Thesaurusbasierte Recherche nach Fachinformationen im Internet

Diplomarbeit

Thema:
Thesaurusbasierte Recherche nach Fachinformationen im Internet

Bearbeiter:
Gerd Klingler

Betreuer:
Dr. Wolf-Fritz Riekert

Fachhochschule Ulm
Fachbereich Technische Informatik
Prof. Dr. rer. nat. Günther Hentschel
Juni 1998

Erklärung

Ich versichere, daß ich die vorliegende Diplomarbeit selbständig und ohne unzulässige fremde Hilfe angefertigt habe.

Die verwendete Literatur ist im Literaturverzeichnis aufgeführt.

Ulm/Donau, Juni 1998

.....
Gerd Klingler

VORWORT

Mein Dank gilt Herrn Prof. Dr. G. Hentschel für die Betreuung dieser Diplomarbeit.

Herr Dr. W.-F. Riekert hat diese Diplomarbeit, im Bereich Umweltinformationssysteme des Forschungsinstituts für anwendungsorientierte Wissensverarbeitung Ulm, ermöglicht. Während dieser Zeit war er als Betreuer ständiger Ansprechpartner für mich. Dafür möchte ich mich recht herzlich bei Ihm bedanken.

Außerdem bedanke ich mich bei allen Personen, die hier nicht namentlich erwähnt werden, mir aber während der Diplomarbeit hilfreich zur Seite standen.

Inhaltsverzeichnis

1 Thesaurusbasierte Recherche nach Fachinformationen im Internet

1.1 Ausgangspunkt

Das World Wide Web (WWW) wird in zunehmendem Maße dazu genutzt, Informationen einem großen Kreis von Nutzern zur Verfügung zu stellen. Entscheidende Bedeutung für die Nutzbarkeit der im WWW angebotenen Informationen sind die Recherchemöglichkeiten, welche den Informationssuchenden zur Verfügung stehen. Grundsätzlich lassen sich dabei folgende Ansätze unterscheiden:

- **Suchmaschinen**
  Suchmaschinen ermöglichen eine Volltextrecherche nach den gesuchten Informationen. Den Suchmaschinen liegt ein Volltextindex, der durch vollautomatische Programme, sogenannte Robots gepflegt wird, zugrunde.
  Die Arbeit der Robots besteht darin, das Internet Tag und Nacht nach neuen und geänderten Einträgen zu durchsuchen. Der Ausgangspunkt ist meist ein Eintrag einer Adresse bei diesen Suchmaschinen ohne zusätzliche Information. Auch ein Querverweis von einer anderen Seite kann zu einem Eintrag führen. Wegen der dabei anfallenden, riesigen Datenmengen werden die gefundenen Seiten mit Hilfe von spezieller Software ausgewertet und nach bestimmten Kriterien in die Datenbank der Suchmaschine eingefügt.
  Beispiel: Altavista, Crawler, Aladin, Fireball, Nathan.
  Vorteil: Suchmaschinen können theoretisch das ganze Internet durchsuchen ohne daß eine zusätzliche manuelle Arbeit geleistet werden muß.
  Nachteil: Eine inhaltliche Interpretation der Suchanfrage ist der Suchmaschine nicht möglich, so daß nur solche Informationsressourcen gefunden werden, die den Suchbegriff wortwörtlich enthalten. Alle anderen Formulierungen führen nicht zum Ziel.
- **Metainformationsserver**
  Metainformationsserver verfügen über einen durchsuchbaren Katalog der Informationsangebote. Die Eintragung von Informationsquellen an die entsprechende Katalogseite wird manuell durchgeführt. Dabei hat der Anbieter der Information die erforderlichen Beschreibungselemente (die ‘Metainformationen‘) i.d.R. beizusteuern.

  Beispiel*:* WWW-UDK des Umweltbundesamtes, Locator des Umweltinformationsnetzes Deutschland.
  Vorteile: Ein gut gepflegter Bestand von Katalogeinträgen steht dem Informationssuchenden zur Verfügung. Häufig führen alternative Formulierungen auch zu den gewünschten Informationen.
  Nachteil: Metainformationsserver erfordern einen hohen Arbeitsaufwand für die Katalogisierung der Informationsangebote.

Metainformationsserver unterstützen ebenso wie Suchmaschinen die Recherche nach Informationen und den direkten Zugriff auf diese Informationen durch Hyperlinks.

1.2 Ziel

Ziel der Diplomarbeit ist es, die Vorteile von Metainformationssystemen und Suchmaschinen zu verbinden. Dem Informationsnutzer wird ein flexibles, problemangemessenes Vokabular für die Recherche an die Hand gegeben, ohne daß dadurch zusätzlicher Aufwand für den Informationsanbieter entsteht. Die Verknüpfung der genannten Vorteile wird durch die Verwendung eines Thesaurus erreicht.
Ein Thesaurus ist ein Nachschlagewerk, welches einen Bezug zwischen dem gesuchten Begriff und dessen Ober- und Unterbegriffen, Synonymen und verwandten Begriffen herstellt. So kann man einem solchen Thesaurus z.B. entnehmen, daß *Kraftstoff* ein Oberbegriff des Begriffs *Benzin* ist.
Ein solcher Thesaurus wird dann mit einer Suchmaschine gekoppelt. Der Thesaurus besitzt für die Recherche zweierlei Funktionen: Zum einen stellen die im Thesaurus enthaltenen Begriffe ein Vokabular an Schlagwörtern dar, das für die Recherche verwendet werden kann. Zum anderen behandelt der Thesaurus diese Begriffe als Entitäten, die über Beziehungen zu einem semantischen Netz verknüpft sind. Im wesentlichen handelt es sich um drei Beziehungstypen, nämlich die Ober-/Unterbegriff-Beziehung, die Beziehung zwischen verwandten Begriffen und die Synonymbeziehung. Darüber hinaus stellen sogenannte multilinguale Thesauri Beziehungen zwischen äquivalenten Begriffen in unterschiedlichen Sprachen dar. Alle diese Beziehungen können für die Interpretation einer Recherche genutzt werden.
Der Informationssuchende wird mit einem einer Suchmaschine vorgeschalteten Thesaurusmodul über eine graphische Benutzeroberfläche interagieren. Dabei wird der Informationssuchende die Möglichkeit haben, in dem problemangemessenen Vokabular des Thesaurus eine Teilstringsuche durchzuführen. Basierend auf diesem Ergebnis kann eine zweite Suche im Thesaurus nach Ober- und Unterbegriffen, sowie auch verwandten Begriffen, Synonymbegriffen und einer entsprechenden Übersetzung erfolgen. Das Ergebnis dieser Interaktion ist eine Suchanfrage, welche als entsprechend parametrisierte URL an die CGI-Schnittstelle der gewählten Suchmaschine gesendet wird.

1.3 Mögliche Anwendungen

1.3.1 Internet

Eine mögliche Anwendung des Systems soll die Vorschaltung des Thesaurusmoduls vor eine allgemeine Suchmaschine im Internet sein. So können für verschiedene Suchgebiete dann verschiedene Thesauri verwendet werden. Prinzipiell ist für jede Art von Informationssuche im Internet die Verwendung eines solchen Thesaurusmoduls möglich. Sinnvoll ist allerdings nur die Verwendung im Zusammenhang mit der Suche nach Fachinformationen, da für allgemeine Informationen die gängigen Suchmaschinen schon riesige Mengen an Informationsquellen liefern. Bei Verwendung eines multilingualen Thesaurus ergibt sich die Möglichkeit, einen Suchbegriff in einer gewünschten Sprache einzugeben, um daraufhin die gewünschte Übersetzung des semantischen Netzes als eigentliche Suchanfrage an die Suchmaschine zu leiten. Hierdurch wird der Thesaurus als Übersetzungsmodul verwendet.

1.3.2 Intranet

Die Verwendung eines solchen Systems innerhalb eines Intranets bietet den Vorteil, daß interne Dokumente nicht mehr katalogisiert werden müssen. Es genügt, eine Suchmaschine mit einem Volltextindex im Intranet zu verwenden und diese Suchmaschine dann unter Verwendung eines Thesaurusmoduls zur Suche im Intranet zu nutzen. Diese sehr kostengünstige Lösung erfordert lediglich eine Suchmaschine mit Volltextindex (z.B. Excite for Webservers) und einen problemangemessenen Thesaurus.

Es besteht die Möglichkeit auf einen eigenen Thesaurus zu verzichten. Dies ist dann möglich, wenn ein Thesaurusmodul, mit dem für die eigenen Zwecke entsprechend geeigneten Thesaurus, im Internet verfügbar ist. Diesem Thesaurusmodul kann eine eigene Suchmaschine im Intranet hinzugefügt werden, so daß als Suchmaschine eine auf dem Intranetserver verwendete Suchmaschine zur Anwendung kommt.

1.4 Anforderungen

Aus den Zielen dieser Diplomarbeit, sowie den daraus resultierenden möglichen Anwendungen eines thesaurusbasierten Suchsystems lassen sich folgende notwendige Systemeigenschaften von vornherein festlegen:

- Implementierung der interaktiven Benutzeroberfläche in der Programmiersprache Java.
- Plattformunabhängige Datenbankanbindung übers Internet.
- Verteilung der Komponenten auf mehrere Rechner.
- Parallele Datenbankzugriffe verschiedener Clients.
- Sichere Datenbankanbindung ans Internet.

2 Die Programmiersprache Java

2.1 Was ist Java

Java ist eine Programmiersprache, die von SUN Microsystems entwickelt wurde. Sie erlaubt es, Anwendungen zu schreiben, die vom Benutzer über das Internet angefordert und auf seiner Maschine ausgeführt werden können, ohne daß der Entwickler die lokale Umgebung des Anwenders, wie Hardware und Betriebssystem, kennen muß. Bestimmte in Java geschriebene Programme, sogenannte Applets, lassen sich insbesondere zur Gestaltung von WWW-Seiten verwenden, die dynamische Elemente, also z.B. bewegte Bilder, enthalten. Java hat innerhalb kurzer Zeit einen regelrechten Boom erlebt. Inzwischen werden zahlreiche Softwareprojekte auf Basis dieser Programmiersprache realisiert.

Java ist

- **einfach**, da auf Zeiger, Zeigerarithmetik, Mehrfachvererbung, Preprozessoranweisung, Überladung von Operatoren, automatische Typkonvertierung, GOTO-Anweisung, C++ Konzepte wie Funktionen, structs und union außerhalb von Klassen verzichtet wurde. Die Syntax von Java entspricht weitgehend der Syntax der Programmiersprachen C und C++. Dadurch ist Java vielen Programmierern in ihrer Darstellungsweise geläufig.
- **objektorientiert**, indem in ihr Erfahrungen aus Eiffel, SmallTalk, C++ münden. Java ist wie diese durch einen relativ kleinen Sprachumfang und umfangreiche Klassen-Bibliotheken (API) gekennzeichnet. Java Programme bestehen aus Klassen. In der Sprache sind wichtige Konzepte der Objektorientierung wie Informationskapselung, Polymorphie, Vererbung, Dynamisches Binden, Nachrichtenaustausch durch Methodenaufruf realisiert.
- **robust**. Programmierer können in der Programmiersprache Java weder direkt Zeiger definieren, noch mittels Zeigerarithmetik auf bestimmte Adressen des Speichers zugreifen. Damit ist eine häufige Fehlerquelle in C- bzw. C++-Programmen beseitigt. Java verwendet statt dessen nur Referenzen auf Objekte. Die Speicherverwaltung übernimmt das Java-Laufzeitsystem, welches nicht mehr referenzierte Objekte aus dem Speicher beseitigt (Garbage Collection). Java ist eine streng typisierte Sprache. In der Sprachspezifikation sind Kompatibilitätsregeln festgelegt, deren Mißachtung zu Fehlern während der Übersetzung oder während der Laufzeit führen. Diese Fehler zwingen zu sauberer Programmierung und verhindern schwer zu findende semantische Fehler, die z.B. in C- bzw. C++-Programmen durch eine automatische Typkonvertierung entstehen können. Die Entwickler von Java haben in die Sprache ein Konzept zur Behandlung von Laufzeitfehlern und Ausnahmen eingeführt. Eine Nichtbeachtung derartiger Ausnahmen im eigenen Programm führt zu Übersetzungsfehlern. Verwendet der Programmierer eine Methode, die eine Ausnahme hervorrufen kann, so muß er dafür sorgen, daß sein Programm im Ausnahmefall ebenfalls weiterarbeiten kann.

- **interpretativ** und besitzt damit die Vorteile interpretierter Sprachen. Seit der Entstehung von Programmiersprachen werden die Vor- und Nachteile interpretierter gegenüber denen compilierter Sprachen gegeneinander abgewogen und führen zu immer neuen Sprachentwürfen. Java verbindet beide Ansätze, indem zunächst der Quelltext in ein Zwischenformat (Java-Byte-Code) compiliert wird und ein Interpreter das Zwischenformat ausführt. Java-Programme sind dadurch schneller als direkt interpretierte Programme und in der Entwicklungsphase leichter zu bearbeiten als compilierte Programme.
- **architekturunabhängig**. Durch die Definition der Zwischenpräsentation wird die Sprache architekturunabhängig, wenn für jede Architektur ein Interpreter zur Verfügung steht.
- **sicher**, das ist eine der wichtigsten Eigenschaften für das beabsichtigte Anwendungsgebiet. Java ermöglicht die Verteilung und Ausführung von Software über ein Netz, insbesondere über das World Wide Web. Ein Mittel, um die dazu benötigte Sicherheit zu erreichen, ist der Byte Code Verifier. Die Übertragung von Java-Programmen erfolgt weder im Quelltextformat noch in einem direkt ausführbaren Format, sondern im Java-Byte-Code-Format. Erst der Java-Interpreter führt Programme dieses Formates aus. Der Byte Code Verifier ist ein Teil des Interpreters, er soll "unkorrekte" Zeilen im Programmtext finden und Laufzeitregeln überprüfen.
- **leistungsstark**. Java erreicht eine hohe Abarbeitungsgeschwindigkeit von Programmen durch die Trennung von Interpreter und Laufzeitumgebung. Mehrere parallel ablaufende Komponenten sorgen für die korrekte Programmabarbeitung. So arbeitet die Komponente, die nicht mehr referenzierte Objekte aus dem Speicher löscht, mit einer niedrigeren Priorität als der Interpreter. Dadurch wird die Abarbeitung des Interpreters beschleunigt.
- **Thread-unterstützend**. Mit der Hilfe von Threads kann der Programmierer parallele Abläufe beschreiben.
- **dynamisch**, sie verwendet dynamisches Binden und dynamisches Laden, sie löst Referenzen zur Superklasse erst im Interpreter auf und beseitigt dadurch die Notwendigkeit zur Neuübersetzung von abgeleiteten Klassen nach der Änderung der Superklasse.
- **Internet-World-Wide-Web-tauglich** durch die Kombination aller oben aufgeführten Eigenschaften.

2.2 Java und die Sicherheit im Internet

Sicherheitsbedenken nehmen eine immer wichtigere Rolle ein, besonders bei Internet-Produkten und Dienstleistung von der elektronischen Verteilung von Software und multimedialen Inhalten bis zum "digitalen Geld". Der Compiler und das Runtime-System haben mehrere Abwehrschichten gegen potentiell inkorrekten Code. Die Java Umgebung geht davon aus, daß nichts und niemandem vertraut werden kann und fährt dementsprechend fort. Dies ist auch im Hinblick auf die Anbindung von Datenbanken ein sehr großer Vorteil. Allerdings gibt es deshalb auch viele Einschränkungen für Java-Applets, also Programme welche im Webbrowser ablaufen.

2.3 Einschränkungen für Applets

Als komplette Sprache ist Java in der Lage jegliche Art von Systemoperationen, welche andere Sprachen können zu verwenden. Diese Möglichkeit ist allerdings nur für Java-Programme gegeben. Aus gutem Grund unterliegen Applets bestimmten Sicherheitsbestimmungen. Erst durch diese Tatsache wurde die rasante Verbreitung von Java überhaupt möglich. Ansonsten wäre es ein leichtes gewesen, unerwünschte Programme (z.B. Viren) auf jeden beliebigen Rechner im WWW zur Ausführung zu bringen.

2.4 Applet-Sicherheitseinschränkungen

Die verschiedenen Webbrowser haben leicht unterschiedliche Einschränkungen für Applets. Bei neueren Webbrowsern ist es auch möglich, Einschränkungen zu lockern. Im allgemeinen ist aber folgendes standardmäßig nicht erlaubt:

- Jeglicher Zugriff auf das lokale Dateisystem.
- Eine Netzverbindung zu einem anderen Computer als dem, von dem das Applet geladen wurde, aufbauen.
- Netzverbindungen auf der lokalen Maschine überwachen oder diese annehmen.
- Ein neues Top-Level-Fenster erzeugen, ohne daß dieses eine deutliche Warnung enthält, daß es nicht vertrauenswürdig ist. Dies hindert Applets daran, andere, vertrauenswürdige Programme vorzutäuschen, in die der Benutzer vielleicht vertrauliche Daten eingeben könnte.
- Daten des Computers, wie z.B. Name des Benutzers, Verzeichnis usw. auslesen.
- Systemeigenschaften definieren.
- Ein Programm auf dem lokalen System aufrufen.
- Den Java-Interpreter beenden.
- Dynamische Bibliotheken des lokalen Systems laden.
- Einen Thread erzeugen oder manipulieren, der nicht zum Applet gehört.
- Einen eigenen SecurityManager erzeugen.
- Netzwerkeigenschaften ändern oder überwachen.

3 Internet-Datenbankanbindung

Der in der letzten Zeit immer stärker werdende Gebrauch des Internets, sowohl im privaten als auch im kommerziellen Bereich, verlangt nach der Möglichkeit eine Datenbank an diese Technologie anzubinden. Außerdem setzen viele Firmen interne Netze auf TCP/IP Basis (sog. Intranets) ein. Diese Technologie erlaubt es, vom Arbeitsplatz aus (unabhängig vom verwendeten System) mit einem Browser auf Firmendaten zugreifen zu können. Diese Entwicklungen haben zur Folge, daß es bei weitem nicht mehr ausreicht, statische Daten auf HTML-Seiten auszugeben. Heutzutage befinden sich bei jeder Firma in einer Vielzahl kommerzieller Anwendungen Datenbanksysteme im Hintergrund. Diese Datenbanken versorgen die Anwendungsprogramme mit den benötigten Informationen (z.B. Produktdatenbanken, Kundendatenbanken, etc.). Schon längere Zeit gibt es verschiedene Möglichkeiten Datenbanken an das Internet/Intranet anzubinden. Man kann dabei grundsätzlich zwischen 2 Arten unterscheiden:

1. **Serverseitiger Zugriff auf die Datenbank:**
   - Common-Gateway-Interface (CGI):
     Der Client sendet über den Webbrowser eine Anfrage an den Webserver. Der Webserver leitet diese Anfrage über das Common-Gateway-Interface an den Application-Server weiter. Dieser stellt eine Verbindung zum Datenbanksystem her, sendet die Anfrage und liefert die Ergebnisdaten wieder über das CGI an den Webserver zurück. Dort wird aus den Daten dynamisch eine HTML-Seite erzeugt und an den Client zurückgeliefert.
     Nachteile: CGI-Programme sind plattformabhängig und datenbankabhängig. Mittels CGI lassen sich nur sehr wenige Standardformen der Interaktivität benutzen. Die Schnittstelle zur Datenbank ist nicht standardisiert.
   - Webserver Application Programming Interface (API):
     Eine der CGI-Methode ähnliche Möglichkeit bietet ein Webserver-API. Bei dieser Methode verwendet der Webserver eine spezielle API für die Verbindung mit dem Application-Server. Dieser stellt wiederum die Anfrage an das Datenbanksystem. Die Übersendung der Ergebnisdaten an den Client geschieht auch hier in Form statischer HTML-Seiten. Diese Möglichkeit bietet eine hohe Performance.
     Nachteile: Die Programmierung der Abfrageroutinen ist komplex. Änderungen der Software (BS, Webserver, usw.) können zu erheblichen Problemen führen, da durch die Verwendung der speziellen API keine System- bzw. Plattformunabhängigkeit gegeben ist.
   - Microsoft® Active Server Pages (ASP)
     Active Server Pages ist eine Skriptumgebung auf Serverseite, in der dynamische, interaktive und leistungsstarke Webserver-Anwendungen erstellt und ausgeführt werden können. Active Server Pages sind programmierbare Webpages die HTML, ODBC, Datenbankzugriffe, usw. ermöglichen, ohne Programmiersprachen wie CGI, JavaScript, Perl und ActiveX verwenden zu müssen. ASP ist nicht von einem bestimmten Webbrowser abhängig, da der Script-Code auf dem Webserver läuft und dieser aus diesem Code HTML-Dokumente erzeugt. Daraus folgt, daß der Browser keine speziellen Eigenschaften besitzen muß.
     Nachteil: Der Webserver muß ASP unterstützen. Dies ist im Moment hauptsächlich nur auf PC-basierten Webservern der Fall.

2. Clientseitiger Zugriff auf die Datenbank:

- Java Database Connectivity (JDBC):
  JDBC stellt eine standardisierte Anwendungsschnittstelle zu relationalen Datenbanksystemen zur Verfügung. Diese Zugriffsmöglichkeiten sind von der Plattform auf der die Anwendung läuft, als auch vom Datenbanksystem unabhängig.

Als die am Besten geeignete Möglichkeit für dieses System, einen Zugriff auf eine Datenbank übers Internet zu realisieren, erweist sich aufgrund der Datenbank- und Plattformunabhängigkeit JDBC.

4 Java Database Connectivity

Seit dem JDK 1.1 ist JDBC Bestandteil der Standarddistribution von Java. Es stellt einen einfachen Mechanismus bereit um mit bestehenden Datenbanken zu kommunizieren. Dabei nehmen Treiber die Schnittstelle zwischen Java-Programm und Datenbank ein. Mit dem gleichen Programmcode kann man also jede beliebige Datenbank ansprechen, sofern dafür ein Treiber existiert.
Ein anderer weitverbreiteter Ansatz ist ODBC (Open Database Connectivity) in der Windows-Welt. JDBC ist, wie der Name es auch schon andeutet, an ODBC angelehnt. Da ODBC jedoch eine C-Schnittstelle darstellt, konnte es nicht einfach exakt übernommen werden, sondern wurde an das objektorientierte Design von Java angepaßt. Viele Datenbank-Hersteller und andere spezielle Anbieter reagierten schnell auf die Möglichkeiten welche durch Java erschlossen wurden mit einem eigenen JDBC-Treiber für ihre Datenbank.

4.1 Treiber

Neben den in der Standardbibliothek enthaltenen Klassen braucht man pro Datenbank einen speziellen Datenbank-Treiber. Er stellt die eigentliche Schnittstelle zwischen dem einheitlichen JDBC und den unterschiedlichen Datenbanken dar. Dieser Treiber kann entweder ein natives Programm sein oder eine Java-Applikation. In den meisten Fällen liefert der Datenbank-Entwickler diese Treiber mit der Datenbank mit. Grundsätzlich unterscheidet man zwischen vier unterschiedlichen JDBC-Treibertypen:

- ODBC Bridge-Treiber
  Treiber dieses Typs übersetzen JDBC-Aufrufe in entsprechende ODBC-Anfragen. Der Einsatz dieses Typ-1-Treibers hat zur Folge, daß ein nativer Code ausgeführt werden muß. Außerdem muß für die Datenbank ein entsprechender ODBC-Treiber installiert sein.
- Native API Partial Java-Treiber
  Diese Typ-2-Treiber übersetzen den JDBC-Aufruf unter Verwendung der herstellerspezifischen Datenbank-API in die Datenbank-API. Hier werden Datenbankmethoden als native Methoden ausgeführt.
- Net Protocol All-Java-Treiber
  Diese Typ-3-Treiber sind vollständig in Java implementiert und werden vom Client geladen. Der Client stellt eine Verbindung zum Server her, welcher wiederum die JDBC-Aufrufe in native Datenbankaufrufe umsetzt.
- Native-Protocol All-Java-Treiber
  Diese Typ-4-Treiber stellen eine direkte Verbindung zur Datenbank her. Sie können sowohl unter Verwendung eines Application-Servers, als auch direkt vom Client aus benutzt werden.

Die Treiber der Typen 1 und 2 ermöglichen unter Ausnutzung bereits bestehender Komponenten einen raschen Zugang zu Datenbanken. Resultierend aus dem Nachteil dieser Treiber, der Verwendung von nativem Code, wird das Java-Projekt einen seiner größten Vorteile, nämlich der Plattformunabhängigkeit beraubt. Da der native Code nicht direkt über das Internet zur Ausführung auf den Client geladen werden kann, ist es notwendig auf jedem Client einen solchen Treiber zu installieren.

4.2 Plattform- und Datenbankunabhängigkeit

Die verschiedenen Datenbanksysteme sind bezüglich spezieller SQL-Statements (Structured Query Language) nicht kompatibel. Dies führt zu folgendem Verhalten der JDBC-API:
JDBC läßt grundsätzlich jede Anfrage zur Datenbank zu, d.h. das SQL-Statement wird von der JDBC-API keiner Überprüfung unterzogen. Die Fehlermeldungen werden dann vom Datenbanksystem zurückgeliefert. Dies hat zur Folge, daß auch syntaktisch falsche Statements an die Datenbank gesendet werden. Fehler werden dann erst vom Datenbanksystem erkannt. Dies kann insbesondere ärgerlich sein, wenn man schon eine größere Tabelle ausgefüllt hat, und sich dann zeigt, daß die Datenbankkonfiguration nicht den Erfordernissen entspricht.
Jede Datenbank hat einen speziellen JDBC-Treiber. Dadurch wird erreicht, daß auch auf spezielle Features einer bestimmten Datenbank nicht verzichtet werden muß.

4.3 JDBC-ODBC-Bridge

Da JDBC zum Zugriff auf eine Datenbank jeweils einen speziellen Treiber benötigt, sah es bei der Veröffentlichung von JDBC mit der Unterstützung für verschiedene Datenbanken schlecht aus. Aus diesem Grund hat SUN einen Treiber entwickelt welcher JDBC-Befehle in ODBC-Befehle umsetzt. Da für die meisten Datenbanken ODBC-Treiber vorhanden sind, können fast alle Datenbanken mit dieser JDBC-ODBC-Bridge angesprochen werden. Da die JDBC-ODBC-Bridge auf C-Bibliotheken zugreifen muß, ist die Verwendung nur in Verbindung mit einem Application Server sinnvoll.

4.4 Funktionsweise von JDBC

JDBC definiert Objekte und Methoden. Mit diesen ist es möglich, daß das Java-Programm auf eine und auch mehrere verschiedene Datenbanken zugreifen kann. Der Zugriff geschieht folgendermaßen:

- Verbindung zur Datenbank über den entsprechenden JDBC-Treiber für das verwendete DBMS herstellen
- Erstellung eines Statement-Objektes
- SQL-Statements über das Statement-Objekt an das darunterliegende DBMS übergeben
- Ergebnisdatensätze abholen
- Verbindung zur Datenbank schließen

Die Java-Applikation läuft dabei normalerweise auf einem Client-Rechner, von welchem aus eine Verbindung zu einem Datenbankserver (oder mehreren) hergestellt wird. Die Datenbankverbindung wird vom JDBC-Treiber-Manager übernommen. Dieser stellt die Verbindung in Form von Java-Objekten zur Verfügung. Der Treiber-Manager ist in der Lage, mehrere Verbindungen zu unterschiedlichen Datenbanken gleichzeitig zu verwalten. Dadurch ist die Möglichkeit gegeben auf verteilte Datenbanken zugreifen zu können.

4.5 Die vier verschiedenen Treiberarchitekturen

4.5.1 Typ-1-Treiberarchitektur

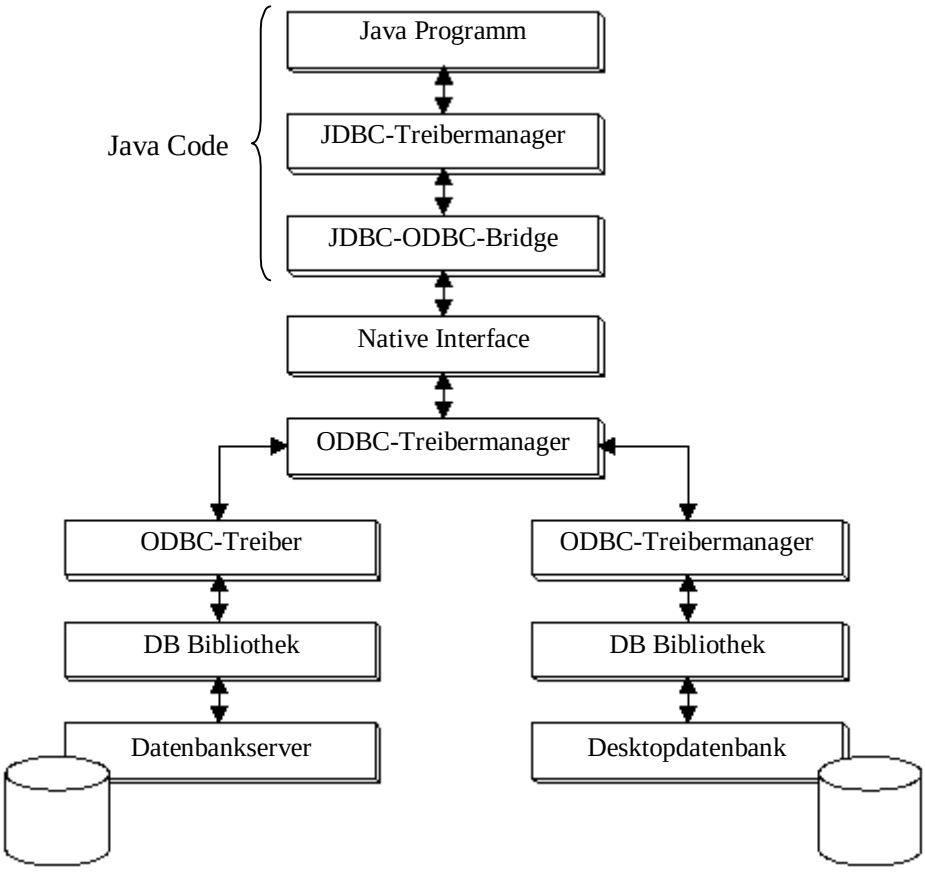

Abbildung 4.1 Typ-1-Treiberarchitektur

Die Java-Applikation bzw. das Applet müssen über den JDBC-Treiber-Manager den JDBC-ODBC-Bridge-Treiber einbinden. Der ODBC-Bridge-Treiber stellt über eine native Schnittstelle (z.B. Windows-DLLs) eine Verbindung zum ODBC-Teiber-Manager her. Vom ODBC-Treiber-Manager wird die Verbindung zur Datenbank über einen lokal installierten ODBC-Treiber hergestellt. Das ganze beruht auf einer Zwei-Ebenen-Architektur, wobei alle Komponenten außer des Datenbankservers beim Client laufen. Handelt es sich um eine Desktop-Datenbank ist das Ganze eine Ein-Ebenen-Architekur.

4.5.2 Typ-2-Treiberarchitektur

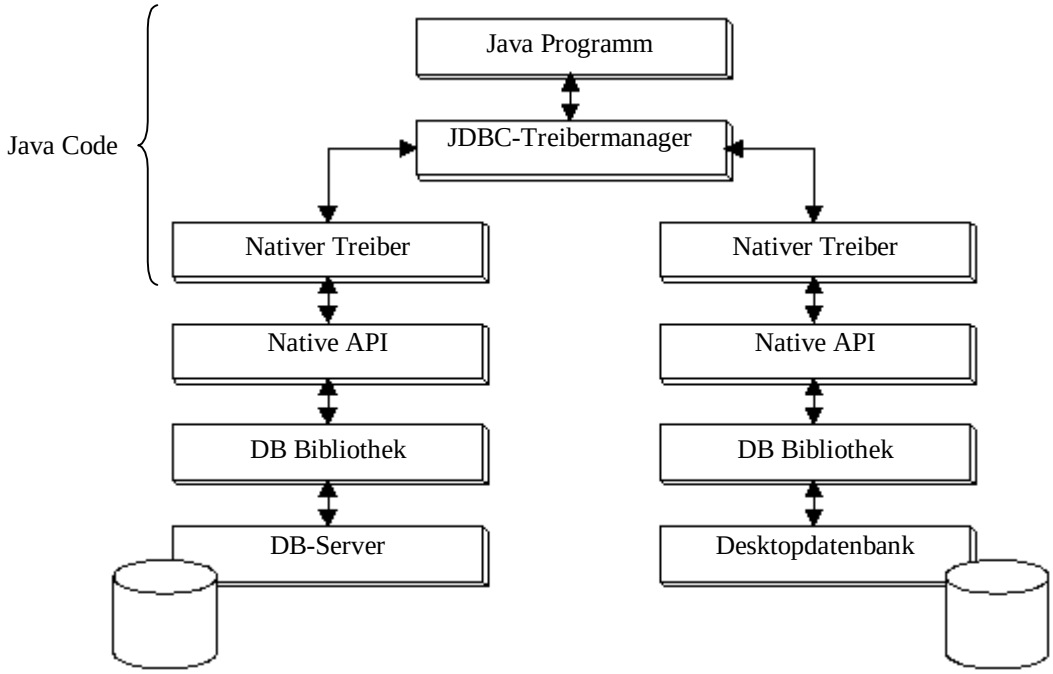

Abbildung 4.2 Typ-2-Treiberarchitektur

Hier handelt es sich ebenfalls um eine Zwei-Ebenen-Architektur. Im Vergleich zu Typ-1-Treibern sind bei diesem Treiber die ODBC-Elemente durch eine Schnittstelle zwischen JDBC-Treiber und nativer Datenbankbibliothek ersetzt worden. Bis auf den Datenbankserver laufen alle Prozesse auf dem Client ab. Ausnahme ist hier wiederum eine Desktopdatenbank.

4.5.3 Typ-3-Treiberarchitektur

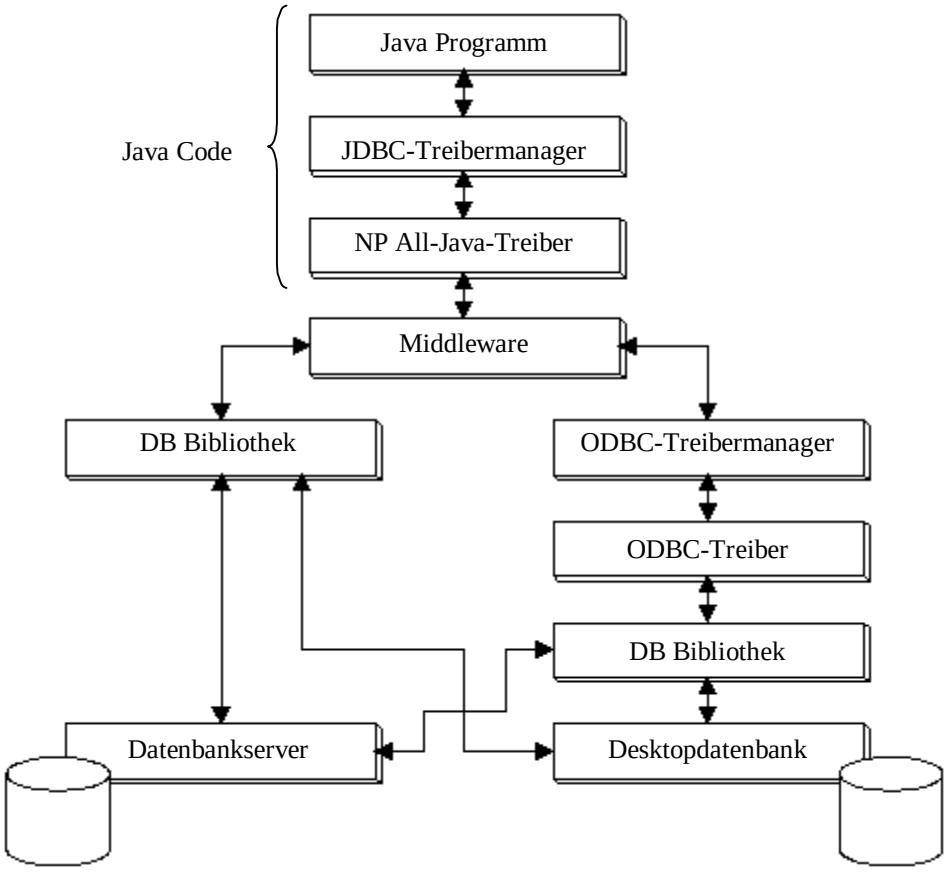

Abbildung 4.3 Typ-3-Treiberarchitektur

Für diesen Typ von Treiber ist eine Middleware zwingend notwendig. Deshalb kann auch auf nativen Code beim Client vollständig verzichtet werden, d.h. zur Laufzeit können alle benötigten Treiber vom Server geladen werden. Die Verbindung zum Datenbankserver kann über ODBC oder aber direkt über datenbankspezifische Treiber stattfinden.

4.5.4 Typ-4-Treiberarchitektur

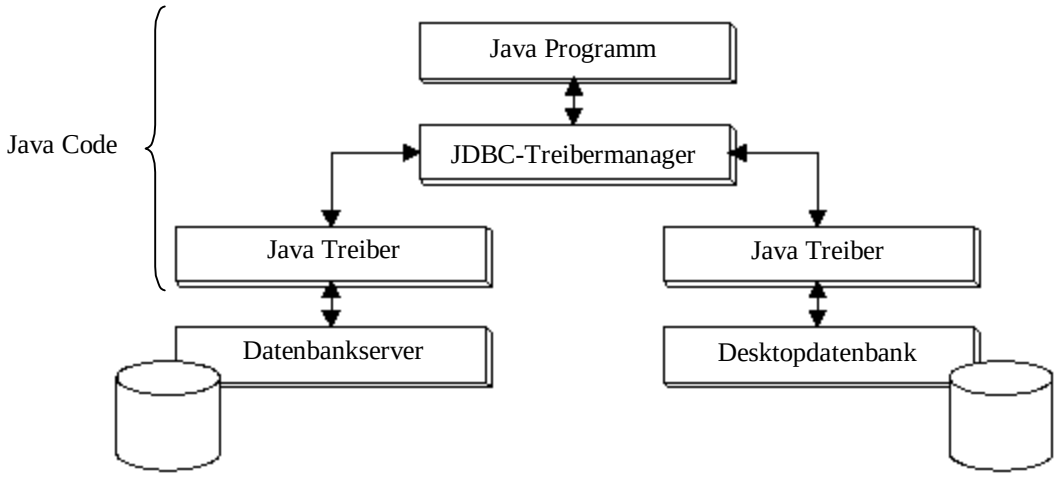

Abbildung 4.4 Typ-4-Treiberarchitektur

Bei JDBC-Treibern des Typs Vier sind weder native, noch ODBC-Treiber notwendig. Der Client kann zur Laufzeit sämtliche benötigte, in Java geschriebene Treiber laden und zur Ausführung bringen. Dies kann man als die modernste Art von JDBC-Treibern bezeichnen.

4.6 Beispiel einer Datenbankabfrage

```
/*
 * Beispiel für einen Datenbankzugriff
 * DBAccess.java
*/


//Notwendige Klassen importieren
import java.sql.*;


class DBAccess {
   // String fuer den Abfrageterm
   public String        query_str     = null;

   // Ergebnissatz
   public ResultSet     result        = null;

   // Name der Tabelle / Datenbank
   public final String  DB_TABLE_NAME = <Name der
Datenbank>

// Name des Benutzers
   public final String  USER_NAME = <Benutzername>

// Passwort des Benutzers
   public final String  USER_PWD  = <Passwort>
```

```
// Meldung 1
   public final String MSG_1 = "Verbindung zur
Datenbank erfolgreich durchgefuehrt."

// Meldung 2
   public final String MSG_2 = "Verbindung zur
Datenbank geschlossen."
// Fehlermeldung 1
   public final String ERR_MSG_1 = "Datenbankfehler: "

// Fehlermeldung 2
   public final String ERR_MSG_2 = "Sonstiger Fehler:
"
   // Main Methode
   public static void main(String args[]) {
      try {
         // JDBC-Treiber laden
         // hier JDBC-ODBC-Bridge Treiber für Access
DB
         Class.forName
("sun.jdbc.odbc.JdbcOdbcDriver");
         // Datenbank-URL
         String db_url = "jdbc:odbc:"+DB_TABLE_NAME;
         // Verbindung zur Datenbank erstellen
         Connection con =
DriverManager.getConnection(db_url, USER_NAME,
USER_PWD);
         System.out.println(MSG_1);
         // Statement erzeugen
         Statement stmt = con.createStatement();
         // Abfrageterm als String festlegen
         query_str = <SQLSTATEMENT>
         // Abfrage ausfuehren
         result = stmt.executeQuery(query_str);
         while(result.next()) {
            // Ergebnis der Spalte 0 auslesen
            // Spalte kann auch über Name angesprochen
werden
            System.out.println(result.getString(0));
         }
         // Verbindung zur Datenbank beenden
         con.close();
         System.out.println(MSG_2);
      }
      // aufgetretene Fehler abfangen
      catch(SQLException sqlex) {
         System.out.println(ERR_MSG_1 +
sqlex.getMessage());
      }
      catch(Exception e) {
         System.out.println(ERR_MSG_2 +
e.getMessage());
      }
```

```
        }
    }
```

5 Architektur eines Gesamtsystems

5.1 Two-Tier-Architekur

Aufgrund der Sicherheitseinschränkungen denen ein Java-Applet unterliegt, kann dieses nur mit dem Rechner in Verbindung treten, von dem es geladen wurde. Deshalb ist es in diesem Fall notwendig, daß der Datenbankserver auf dem gleichen Rechner wie der Webserver installiert ist. Dies nennt man die "Two-Tier-Architektur". Diese Architektur entspricht der allgemeinen Client-Server-Architektur.

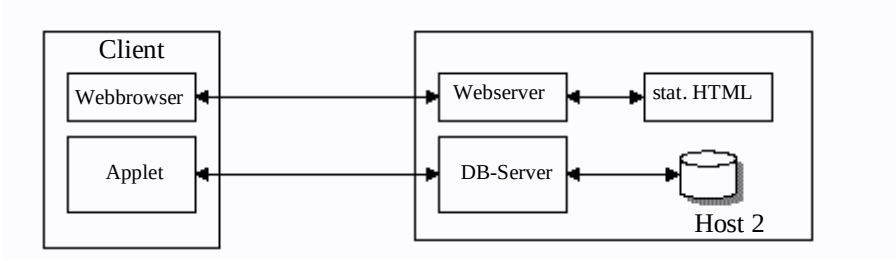

Abbildung 5.1 Two-Tier-Architektur

5.2 Three-Tier-Architektur

Sind der Webserver und der Datenbankserver auf getrennten Maschinen installiert, so spricht man von der "Three-Tier-Architektur". Hierbei wird ein Applet vom Webserver geladen. Das Applet kommuniziert mit einem Application-Server oder einem Gateway (z.B. CGI) auf dem Webserver. Der Application-Server bzw. das Gateway fungieren dann als DB-Client und stellen die Anfrage an die Datenbank. Die Datenbank liefert das Ergebnis an den DB-Client und von dort aus wird das Ergebnis zum Applet weitergeleitet.

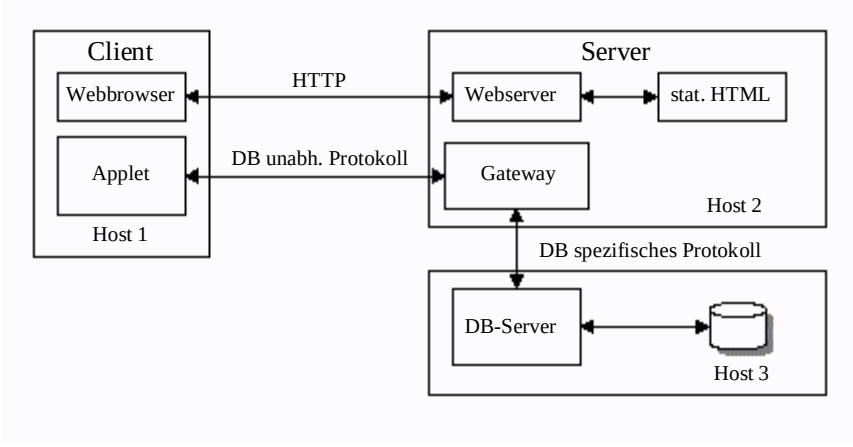

Abbildung 5.2 Three-Tier-Architektur mit Gateway

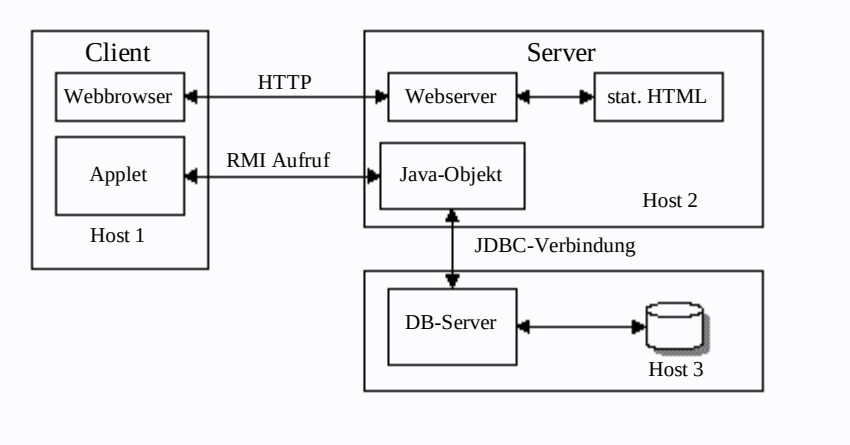

Abbildung 5.3 Three-Tier-Architektur mit Java RMI

5.3 Sicherheit

Eine wichtige Rolle spielt die Frage der Sicherheit von Daten auf einem Datenbanksystem. Viele Datenbanken enthalten sensible Daten, welche aus den verschiedensten Gründen vor dem Zugriff nicht authorisierter Personen geschützt werden müssen. Die Frage der Sicherheit von Datenbanken kann innerhalb des Intranets durch die Sicherheitsmechanismen der Datenbank gelöst werden, so können z.B. nur bestimmte IP-Adressen für einen Zugriff zugelassen werden. Die Möglichkeiten sind aber nach der Anbindung der Datenbank ans Internet nicht mehr ausreichend. Aus diesem Grund werden z.B. Firewalls eingesetzt.

5.3.1 Firewall

Eine Firewall versucht eine bestehende Sicherheitspolitik umzusetzen. Firewalls können auf unterschiedliche Weise konfiguriert werden, beispielsweise so, daß sie nur bestimmte Dienste zulassen. Generell sind Firewalls so konfiguriert, daß sie nur authorisierte Zugriffe zulassen. Eine Firewall kann nur dann ihre Aufgabe erfüllen, wenn es keine Möglichkeit gibt, zwischen den beteiligten Netzen einen Datenverkehr unter Umgehung der Firewall durchzuführen. Die Firewall kontrolliert also den Übergang zwischen zwei Netzen.

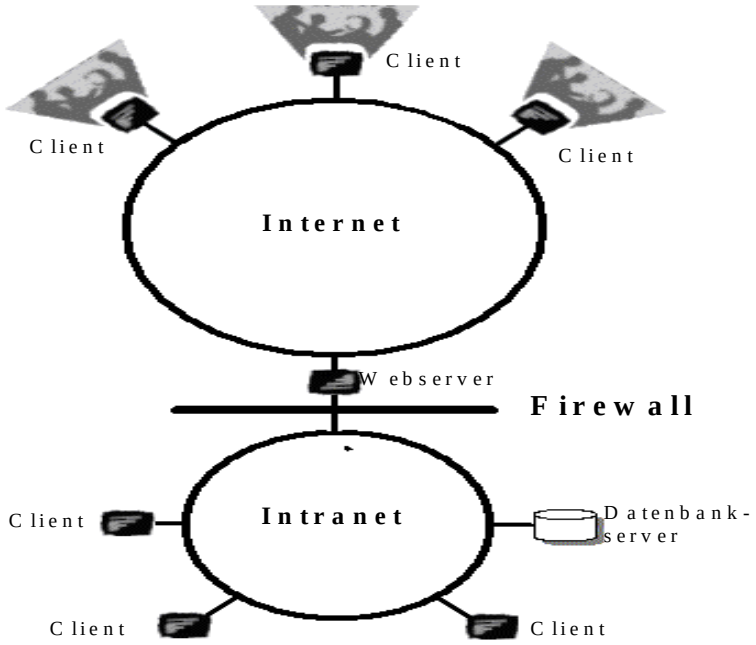

Abbildung 5.4 Firewall

5.3.2 Java Security-API

Die Java-Security-API erlaubt es, abgestufte Sicherheitsmechanismen in Java-Programme zu integrieren. Die wohl wichtigsten Möglichkeiten in dieser API sind Digitale Signaturen, Schlüsselmanagment und Zugriffskontrolle. Außerdem können sogenannte *Trusted Applets* erzeugt werden. Hierzu wird ein JAR-File (Java Archive File) signiert. Man kann nun einem dieser signierten Applets mehr Rechte geben, als standardmäßig vorgesehen sind. Dadurch kann man einem Applet z.B. erlauben, daß es Daten auf der lokalen Platte abspeichert um diese bei einem erneuten Start z.B. als Standardwerte übernimmt. Durch verschiedene Verschlüsselungsmechanismen kann ein hoher Stand an Sicherheit im Internet erreicht werden.

5.3.3 Verschiedene Ebenen der Authentifizierung

Eine Möglichkeit der Authentifizierung besteht darin, den Zugang zum Webserver über ein Paßwort zu schützen. Diese Möglichkeit ist jedoch nur für wenige Anwendungen sinnvoll, da das Laden eines HTML-Dokumentes und/oder eines Applets nicht bedeutet, daß ein Zugriff auf Daten in der Datenbank erfolgen soll. Durch diese Methode würden aber viele unnötige Paßwortabfragen notwendig.

Eine andere Möglichkeit ist es, die Paßwortabfrage an der Datenbank durchzuführen. Schlägt dies fehl, so wird die Fehlermeldung der Datenbank an das Java-Applet übermittelt. Hierauf reagiert das Applet mit einer Fehlerroutine, so daß der Benutzer über seine fehlerhafte Anmeldung informiert wird.

Bei der Three-Tier-Architektur kann die Authentifizierung stattfinden, wobei der eigentliche Client keine Verbindung zur Datenbank benötigt. In diesem Fall wird die Passwortabfrage durch den Application-Server durchgeführt. Dieser kann dann im Fall eines bestehenden Accounts die gewünschten Daten auslesen, andernfalls entsprechend reagieren. In diesem Fall ist die Datenbank durch den Application-Server vollständig von der Außenwelt abgenabelt. Durch die Verwendung von Sicherheitsmechanismen zwischen Applet und Application-Server kann diese Verbindung als sichere Verbindung bezeichnet werden. Zusätzlich können zwischen Application-Server und Datenbank zusätzliche Sicherheitsmechanismen installiert werden. Hierdurch sollte ein System mit höchsten Sicherheitsanforderungen zu realisieren sein.

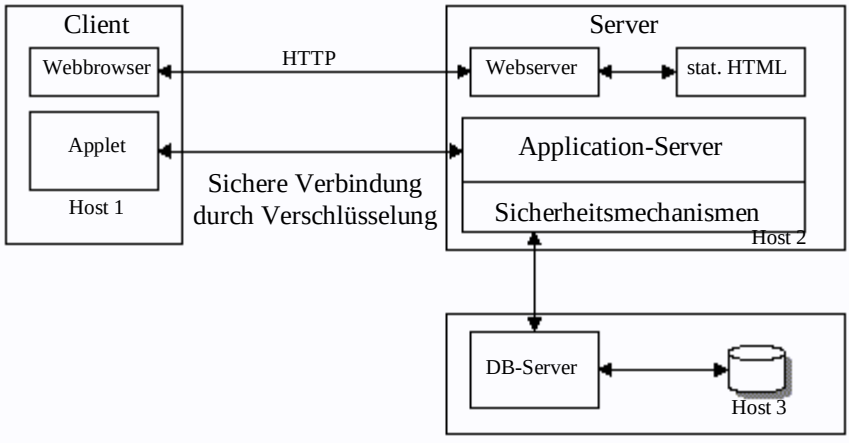

Abbildung 5.5 Authentifizierung

6 Verteilte Systeme

6.1 Die wichtigsten Standards

6.1.1 CORBA

CORBA (Common Object Request Broker Architecture) unterstützt die Entwicklung und Integration objektorientierter Software-Komponenten in verteilten, heterogenen Umgebungen. CORBA ist von der OMG (Object Management Group) standardisiert. Durch den Object Request Broker wird die Kommunikations-Infrastruktur zur Verfügung gestellt. Dadurch können Operationen auf entfernten Objekten ausgeführt werden. Der Aufruf einer Operation ist transparent bezüglich Hard- und Software-Plattform, Datenrepräsentation, Programmiersprache, Netzwerk und Transport.

6.1.2 DCOM

DCOM (Distributed Component Object Model) ist die verteilte Variante von COM. Komponenten können nicht nur lokal auf einer Maschine zum Einsatz kommen, sondern es können von einem Client auch entfernte Komponenten angesteuert werden. Die Verteilung basiert auf Microsoft RPC. Mit DCOM werden die Funktionen einer Komponenten über einheitliche Schnittstellen den Clients zur Verfügung gestellt. Die in COM verwendete Technologie ist technisch gesehen verwandt mit CORBA; Schnittstellen, Protokoll und Entwicklungswerkzeuge sind aber inkompatibel. Die beiden Welten können aber heute relativ gut integriert werden, da von einigen Herstellern CORBA<->DCOM Gateways angeboten werden.

6.1.3 OSF/DCE

OSF/DCE (Open Software Foundation/Distributed Computing Environment) ist speziell für die Entwickung C-basierter Applikationen in verteiltem Umfeld gedacht. Die DCE-Architektur definiert den DCE Executive und die DCE Extended Services.
Der Remote Procedure Call Mechanismus des DCE Executive (RPC) erlaubt Clients Funktionen auf entfernten Servern aufzurufen. Ein Client kann hierbei einen Server mittels des Directory Services suchen und zu ihm binden. Durch den RPC-Mechanismus wird der Client von kommunikationsspezifischen Details abgeschirmt.

6.1.4 RMI

Java RMI (Remote Method Invocation) ist die SUN spezifizierte Variante für die die Entwicklung verteilter Java-Applikationen. RMI ist ausschließlich für die Sprache Java verfügbar. Dies bietet den Vorteil, daß damit Java-spezifische Möglichkeiten ausgenützt werden können. Der Nachteil ist allerdings, daß RMI auf die Sprache Java beschränkt bleibt.
RMI basiert auf RPC, legt aber eine objektorientierte, in Java implementierte Schicht über RPC. Damit ergeben sich gegenüber der herkömmlichen RPC-Programmierung starke Vereinfachungen. So werden z.B. Interfaces in der Sprache Java spezifiziert.

7 Remote Method Invocation (RMI)

RMI ist ab der JDK Version 1.1 Teil der Core-API, d.h. standardmäßig im Package java.rmi enthalten. Für die frühere Version 1.0.2 des JDK gibt es allerdings eine eigene Version von RMI. Durch RMI wird Java-Programmen die Möglichkeit gegeben, untereinander auch in verschiedenen Adreßräumen zu kommunizieren. RMI geht im Vergleich zu CORBA von einer homogenen Umgebung aus, da es nicht wie CORBA ein sprachunabhängiges Objektmodell voraussetzt. RMI ist gegenüber CORBA auch bezüglich der Geschwindigkeit im Nachteil, da RMI HTTP verwendet um Klassen zu laden, während CORBA das schnellere IIOP-Protokoll verwendet.

7.1 Verteilte Objekte

Als verteiltes Objekt bezeichnet man ein Objekt dessen Methoden von einer anderen virtuellen Java-Maschine aus aufgerufen werden können. Verteilte Objekte müssen eine spezielle Schnittstelle (Interface) implementieren. Alle Methoden welche von anderen Programmen mittels RMI aufgerufen werden können, müssen in diesem Interface deklariert sein.

7.2 Aufruf einer Methode

Der Aufruf einer Methode eines sich in einem anderen Adreßraum befindlichen Objektes erfolgt über eine Referenz.

7.3 Namensdienst

Ein Namensdienst übernimmt die Zuordnung zwischen Namen und Objektreferenzen. Benötigt ein Client die Referenz auf ein verteiltes Objekt, kann er diese über den Namensdienst erhalten, sofern dieses Objekt im Namensdienst eingetragen wurde. Der Client muß hierfür lediglich den Namen kennen.

7.4 Stubs und Skeletons

Stubs und Skeletons sind Objekte, welche dem Server zur Kommunikation mit dem Client, bzw. dem Client als Ersatz für die eigentliche Implementierung des Interfaces dienen.

7.5 Beispiel einer verteilten Anwendung

Um eine verteilte Anwendung in Java mittels RMI zu realisieren sind mindestens zwei Klassen und ein Interface zu schreiben. Im Interface müssen allen Methoden, welche „remote“ ausgeführt werden sollen, deklariert sein.
Der Application-Server muß diese Methoden des Interfaces implementieren. Diese vom Interface deklarierten Methoden können dann vom Client aufgerufen werden.

7.5.1 RMIServerImpl

```
/*
 * Interface für eine einfache RMI Anwendung
 * RMIServerImpl.java
*/

public interface RMIServerImpl.java extends
java.rmi.Remote {

   // Deklaration der remote Methode

   String sayHello() throws java.rmi.RemoteException;

}
```

7.5.2 RMIServer

```
/*
 * Programm für eine einfache RMI Anwendung
 * RMIServer.java
/

import java.rmi.*;
import java.rmi.server.UnicastRemoteObject;

public class RMIServer.java extends
UnicastRemoteObject implements RMIServerImpl {

   private String name         = null;
   static  String DEAFULT_NAME = "RMIServer";
   static  int    PORT         = 1099;

   // Konstruktor der Klasse

   public RMIServer(String name) throws
RemoteException {
      super();
      this.name = name;
   }

   // Methode welche remote aufgerufen wird;
   public String sayHello() throws Remote Eception {
      return "Hello";
   }

   // Methode main, wird am Anfang aufgerufen
   public static void main(String args[]) {
      // Sicherheitsmanager erzeugen
      System.setSecurityManager(new
RMISecurityManager());
      try {
         System.out.println("Try to create Application
Server");
         // starten der Registry auf speziellem Port

java.rmi.registry.LocateRegistry.createRegistry(port);
         // erzeugen einer Instanz durch aufruf des
Konstruktors
         RMIServer rs = new RMIServer(DEFAULT_NAME);
         // Registrieren des Servers
         Naming.rebind(DEFAULT_NAME, obj);
         System.out.println("RMIQuery created");
      }
      catch(Exception e) {
         // Ausgabe des aufgetretenen Fehlers
         System.out.println("main: an exception
occured!");
         e.printStackTrace();
```

```
        }
    }
```

7.5.3 RMIApplet

```
/*
 * Applet für eine einfache RMI Anwendung
 * RMIApplet.java
/
import java.applet.*;
import java.awt.*;

public class RMIApplet extends Applet {

   RMIServerImpl remote_object = null;

   // Konstruktor des Applets
   public void RMIApplet()
      super();
   }
   // Initialisierung des Applets
   public void init() {
      try {
         // Remote Objekt aus Registry ermitteln
         remote_object =
(RMIServerImpl)Naming.lookup("//"+getCodeBase().getHos
t()+"/RMIServer");
        // remote Methode aufrufen
        System.out.println(remote_object.sayHello()+"
World");
      }
      catch(Exception e) {
         // Fehlerbehandlung
         ...
      }
   }
}
```

Wenn es sich um ein Programm handelt, muß auch beim Client ein Security-Manager installiert werden. Bei Applets übernimmt dies der Webbrowser.

7.5.4 Ausführung

Für das Applet muß ein HTML-Dokument erzeugt werden.
Alle drei Programme müssen kompiliert werden.

```
javac *.java
```

Es ist notwendig, für den Client und den Server die sogenannten Stubs und Skeletons zu erzeugen. Dies geschieht mit dem RMI-Compiler RMIC.

```
rmic RMIServerImpl
```

Nun kann die Anwendung gestartet werden. Der RMIServer mit dem JRE, das Applet mit einem Webbrowser oder dem Appletviewer.

```
java RMIServer
appletviewer <HTML-DOKUMENT>
```

Falls die RMIRegistry nicht im RMIServer gestartet wird, muß vor der Ausführung der Programme die Registry gestartet werden.

```
rmiregistry
```

7.6 Object Serialization

7.6.1 Erlaubte Datentypen für RMI-Methoden

Als Rückgabewert von RMI-Methoden sind nur Datentypen zugelassen, welche das Interface *java.io.Serializable* implementieren. Diese Schnittstelle sorgt dafür, daß ein Objekt in seinem aktuellen Zustand in einen Stream geschrieben und auch exakt so wieder ausgelesen werden kann. Alle Felder der Klasse werden geschrieben und beim Empfänger wieder eingelesen. Standardmäßig implementieren nicht alle verfügbaren Datentypen die Schnittstelle *java.io.Serializable*. So wird dieses Interface auch von der Klasse *java.sql.ResultSet*, dem Ergebnistyp einer SQL-Abfrage nicht implementiert.

7.6.2 Beispiel Serialization

```
/*
 * Beispiel wie eine Klasse aussehen muß,
 * damit sie mit RMI übertragen werden kann
 * Die Klasse java.awt.Canvas implementiert die
 * Schnittstelle java.io.Serializable bereits, deshalb
 * müssen die Felder dieser Klasse nicht übertragen
werden
 * SerExample.java
*/


import java.io.;
import java.awt.*


public class SerExample extends java.awt.Canvas
implements java.io.Serializable {

   // Felder die übertragen werden müssen
   public int          id   = 0;
   public String       name = null;

   // Rekursives Datenfeld
   public SerExample   re   = null;



   // Konstruktor der Methode
   public SerExample(int id, String name) {
      this.id   = id;
      this.name = name;
      if(id-1 > 0) re = new SerExample(--id, name);
   }


   // Daten Einlesen
   private void readObject(java.io.ObjectInputStream
```

```
in) throws IOException, ClassNotFoundException {
      id   =
Integer.parseInt(((String)in.readObject()));
      name = (String)in.readObject();
      re   = (SerExample)in.readObject();
   }


   private void writeObject(java.io.ObjectOutputStream
out) throws IOException, ClassNotFoundException {
      out.writeObject(Integer.toString(id));
      out.writeObject(name);
      out.writeObject(re);
   }
}
```

8 Beispielrealisierung des Systems

Unter Verwendung des zweisprachigen Umweltthesaurus des Umweltbundesamtes entstand ein Prototyp dieses Systems. Der verwendete Thesaurus liegt am FAW Ulm in Form einer relationalen Datenbank entsprechend den Normen ISO 2788 und ISO 5964 vor. Um die Portierbarkeit des Systems zu demonstrieren, wurden zwei verschiedene Varianten getestet. Ein Oracle7 Datenbankserver unter UNIX und eine Access-Desktopdatenbank unter WIN NT. Um das Programm für beide Plattformen zu benutzen, muß lediglich der JDBC-Treiber umgestellt werden. Das System ist als Three-Tier-Architektur konzipiert, da aber im Fall der Desktopdatenbank der Webserver auf der gleichen Maschine wie das Datenbanksystem läuft, wird damit auch die vielfältige Einsatzfähigkeit dieser Systemarchitektur gezeigt.

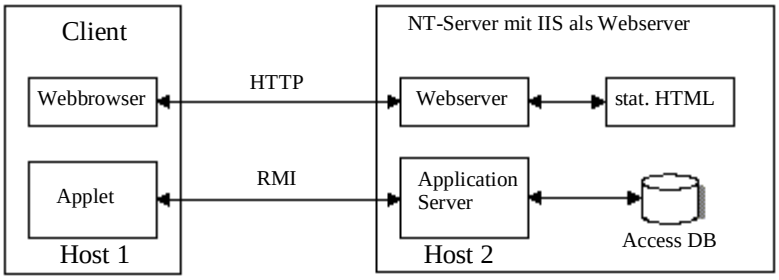

Abbildung 8.1 System mit Access-Datenbank

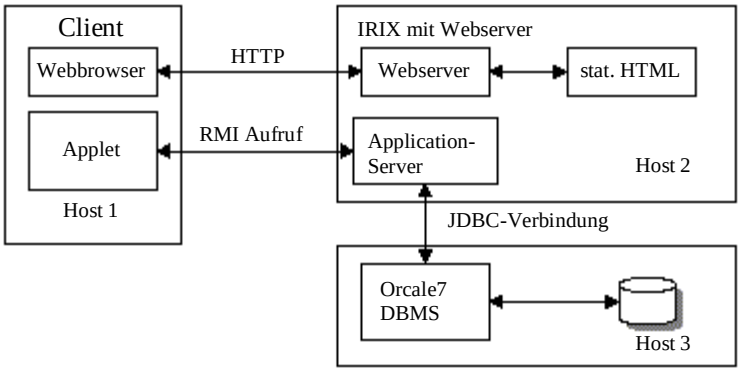

Abbildung 8.2 System mit Orcale-Datenbank

8.1 Die Thesaurusdatenbank

8.1.1 Die Tabellenstrukturen

Struktur der Tabelle lang_table:

Attributname	Datentyp	Bedeutung
Language	Char(2)	Nummer die eine Sprache identifiziert
Mnemonic	Char(20)	Sprache

In dieser Tabelle wird jeder Sprache eine eindeutige Nummer zugeordnet. Diese Nummer gilt als Referenzierung der Sprache des jeweiligen Thesaurusbegriffs in den folgenden Tabellen.

Struktur der Tabelle ThesDesc:

Attributname	Datentyp	Bedeutung
Desc_No	Number(38)	Automatisch generierter Schlüssel
Language	Char(2)	Nummer die eine Sprache identifiziert
Term	Char(60)	Deskriptor

In dieser Tabelle werden die eigentlichen Thesaurusbegriffe (die Deskriptoren) gespeichert. Der Primärschlüssel besteht aus den Attributen *Language* und *Desc_No*. Begriffe gleicher Bedeutung in unterschiedlichen Sprachen sind unter der gleichen *Desc_No* abgelegt. Sie unterscheiden sich anhand des Attributs *Language*.

Struktur der Tabelle ThesNar:

Attributname	Datentyp	Bedeutung
Desc_No	Number(38)	Automatisch generierter Schlüssel
Language	Char(2)	Nummer die eine Sprache identifiziert
Nar_Desc_No	Number(38)	Desc_No des Unterbegriffs

Hier wird die Verbindung zwischen Deskriptoren und deren Unterbegriffen festgelegt. Unter- wie Oberbegriffe sind Deskriptoren. Jeder Deskriptor ist Oberbegriff zu seinen Unterbegriffen. Es reicht deshalb die Verbindung in eine Richtung.

Struktur der Tabelle ThesSyn:

Attributname	Datentyp	Bedeutung
Desc_No	Number(38)	Automatisch generierter Schlüssel
Language	Char(2)	Nummer die eine Sprache identifiziert
Synonyme	Char(70)	Zugehöriger Synonymbegriff

In dieser Tabelle werden einem Deskriptor Synonyme zugewiesen. Zu einem Deskriptor können mehrere Synoyme eingetragen sein, da sich der Primärschlüssel aus allen drei Attributen zusammen setzt.

Struktur der Tabelle ThesRel:

Attributname	Datentyp	Bedeutung
Desc_No	Number(38)	Automatisch generierter Schlüssel
Language	Char(2)	Nummer die eine Sprache identifiziert
Related_Term	Char(70)	Verwandter Begriff

Hier wird die Verbindung zwischen einem Deskriptor und seinen verwandten Begriffen festgelegt. Zu einem Deskriptor können mehrere verwandte Begriffe eingetragen sein, da sich der Primärschlüssel aus allen drei Attributen zusammen setzt.

Struktur der Tabelle ThesSco:

Attributname	Datentyp	Bedeutung
Desc_No	Number(38)	Automatisch generierter Schlüssel
Language	Char(2)	Nummer die eine Sprache identifiziert
Scope_Note	Char(70)	Erläuterung zum Begriff

In dieser Tabelle sind Erläuterungen zu einem Deskriptor abgelegt. Zu einem Deskriptor können mehrere Erläuterungen eingetragen sein, da sich der Primärschlüssel aus allen drei Attributen zusammen setzt.

8.1.2 Die Relationen der Thesaurusdatenbank

Die 4 verschiedenen Arten der Beziehungen sind:

- 1:1 Ein Wert einer ersten Entitätsmenge ist genau einem Wert einer zweiten Entitätsmenge und umgekehrt zugeordnet.
- 1:n Ein Wert einer Entitätsmenge ist mehreren Werten einer zweiten Entitätsmenge zugeordnet. Umgekehrt ist einem Wert der zweiten Entitätsmenge nur ein Wert der ersten Entitätsmenge zugeordnet.
- n:1 Einem Wert der ersten Entitätsmenge ist nur ein Wert der zweiten Entitätsmenge zugeordnet. Einem Wert der zweiten Entitätsmenge sind mehrere Werte der ersten Entiätsmenge zugeordnet.
- m:n Einem Wert der ersten Entitätsmenge sind mehrere Werte der zweiten Entitätsmenge und umgekehrt zugeordnet.

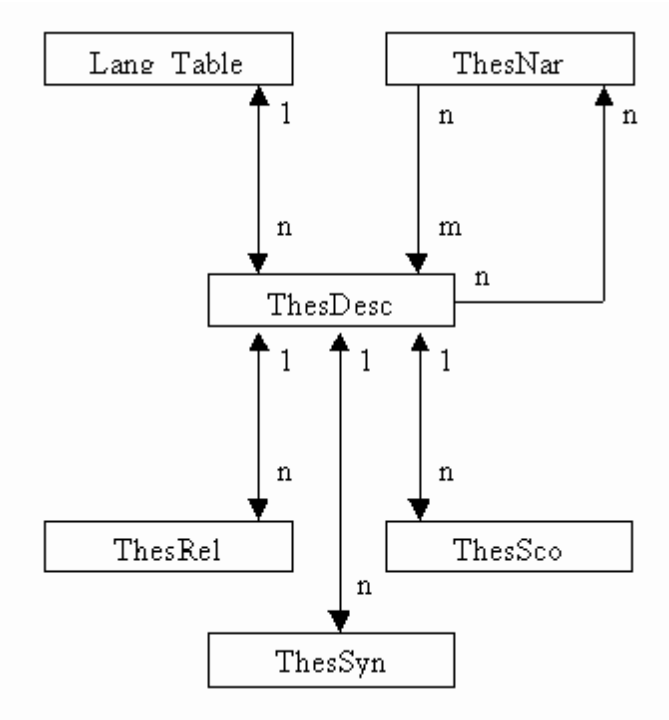

Abbildung 8.3 Relationen der Thesaurusdatenbank

8.2 SQL-Statements

Um möglichst viele verschiedene Datenbanken mit dem Thesaurusmodul ansprechen zu können, sind zur Abfrage von Datensätzen nur standardmäßige SQL-Statements, und keine speziellen, datenbankabhängige Befehle verwendet worden. Dies kann zwar zu Verlusten bezüglich der Geschwindigkeit führen, man kann aber ohne Probleme eine andere Datenbank anbinden. Nur deshalb ist es auch möglich, sowohl die Access- als auch die Oracle-Datenbank, ohne eine Änderung der SQL-Statements, anzusprechen.

8.3 Verteilung der Komponenten

Alle Datenbankabfragen werden mittels RMI-Methoden durchgeführt. Die Datensätze werden vor der Rückgabe an den Client so aufbereitet, daß für den Client nur noch geringe Arbeit notwendig ist die Daten graphisch darzustellen. Dieser Ansatz der Verteilung bietet den Vorteil, daß nur tatsächlich vom Client benötigte Daten übertragen werden müssen und so die Netzwerkbelastung möglichst gering ist.

8.4 Die Betriebssysteme

8.4.1 Clients

Für die Clients sind alle möglichen Betriebssysteme denkbar, sofern es für das jeweilige Betriebssystem einen Webbrowser mit Java-Interpreter gibt.

8.4.2 Application-Server

Als Plattform für den Application-Server sind alle Betriebssysteme denkbar, sofern es für diese ein Java Laufzeitsystem gibt. In der Zwischenzeit gibt es für eine Vielzahl von Plattformen Portierungen von Java, so daß nahezu jede Plattform denkbar ist.

8.5 Funktionsweise des Systems

Das Applet versucht beim Start die Sprachunterstützung des Thesaurus abzufragen. Dazu wird eine Methode des Application-Servers aufgerufen. Diese Methode fragt bei der Datenbank die Sprachuntersützung des Thesaurus ab. Der Benutzer des Thesaurusmoduls kann nun einen Begriff innerhalb des Thesaurus suchen. Die Suchmethode entspricht einer Teilstringsuche welche alle Begriffe, Synonyme und verwandte Begriffe mit Verweisen auf den jeweiligen Deskriptor, als Ergebnis liefert. Durch eine zweite Anfrage erhält der Informationssuchende eine graphische Darstellung des semantischen Netzes bezüglich eines Begriffe. Die Suchanfrage für die Suchmaschine kann nun aus diesen Begriffen zusammengestellt werden.

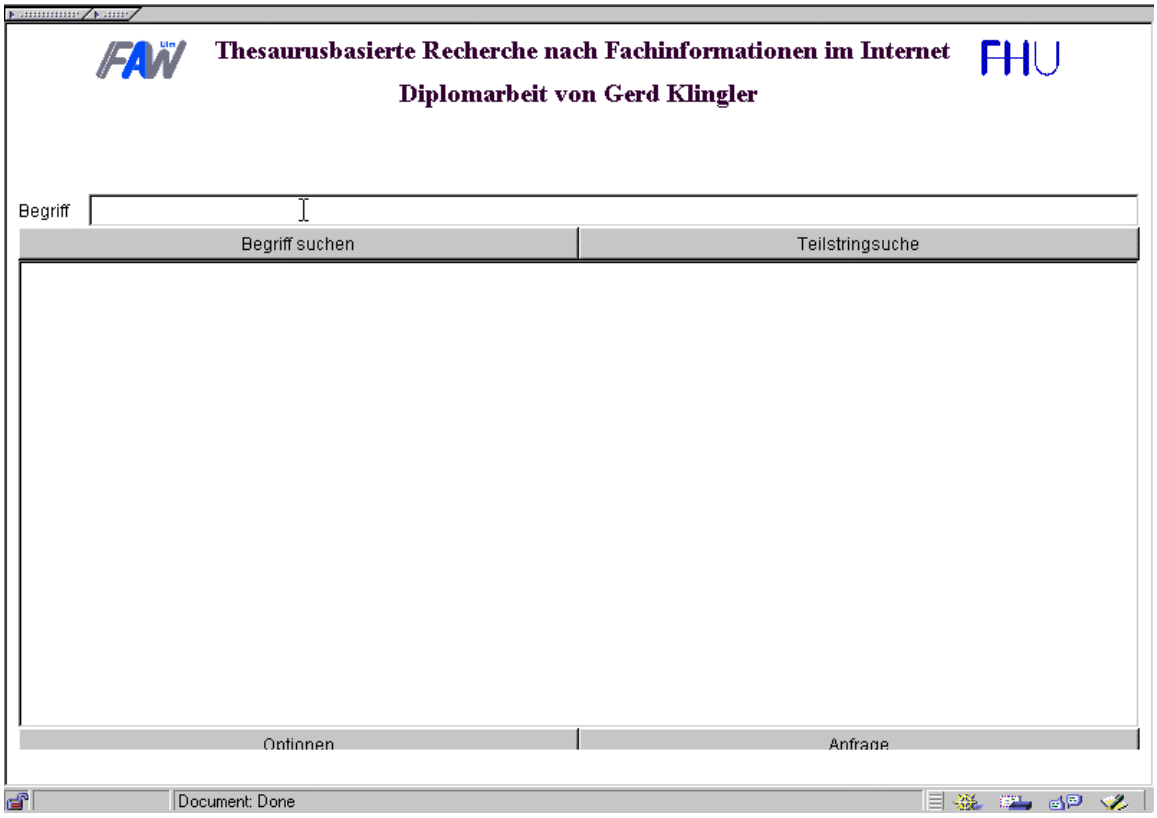

Abbildung 8.4 Benutzeroberfläche des Thesaurusmoduls nach dem Start

Verschiedene Parameter können bezüglich der Suche im Thesaurus eingestellt werden. So können die Hierarchiestufen für Ober- bzw. Unterbegriffe eingegeben werden. Diese beiden Parameter sind zeitlich relevant für die Datenbankanfrage. Die anderen Parameter sind für die Datenbankanfrage weniger relevant, deswegen werden diese auch immer mit ausgeführt, aber nur auf Wunsch auch angezeigt.

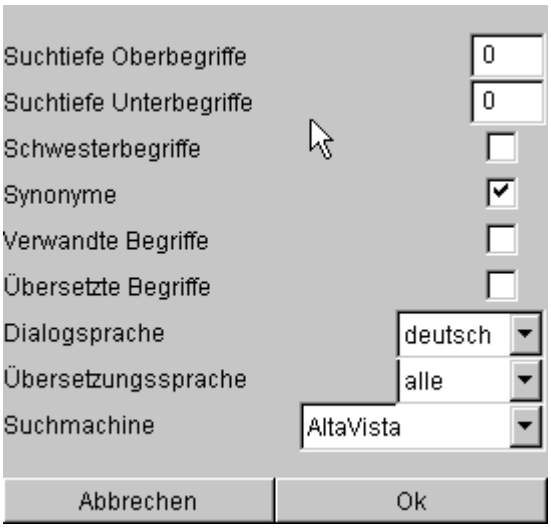

Abbildung 8.5 Mögliche Optionen des Thesaurusmoduls

Als Ergebnis einer ersten Anfrage an den Thesaurus erhält man eine Liste mit den entsprechenden Einträgen. Bei der Suche im Thesaurus werden alle Begriffe gefunden, welche mit dem entsprechenden Begriff beginnen. Dies entspricht einer Teilstringsuche. Bei der Suche werden auch die Tabellen mit den verwandten Begriffen und Synonymen durchsucht. Entspricht dort ein Eintrag dem Suchmuster, so wird der Begriff mit einem entsprechenden Verweis auf den Deskriptor zurückgeliefert.

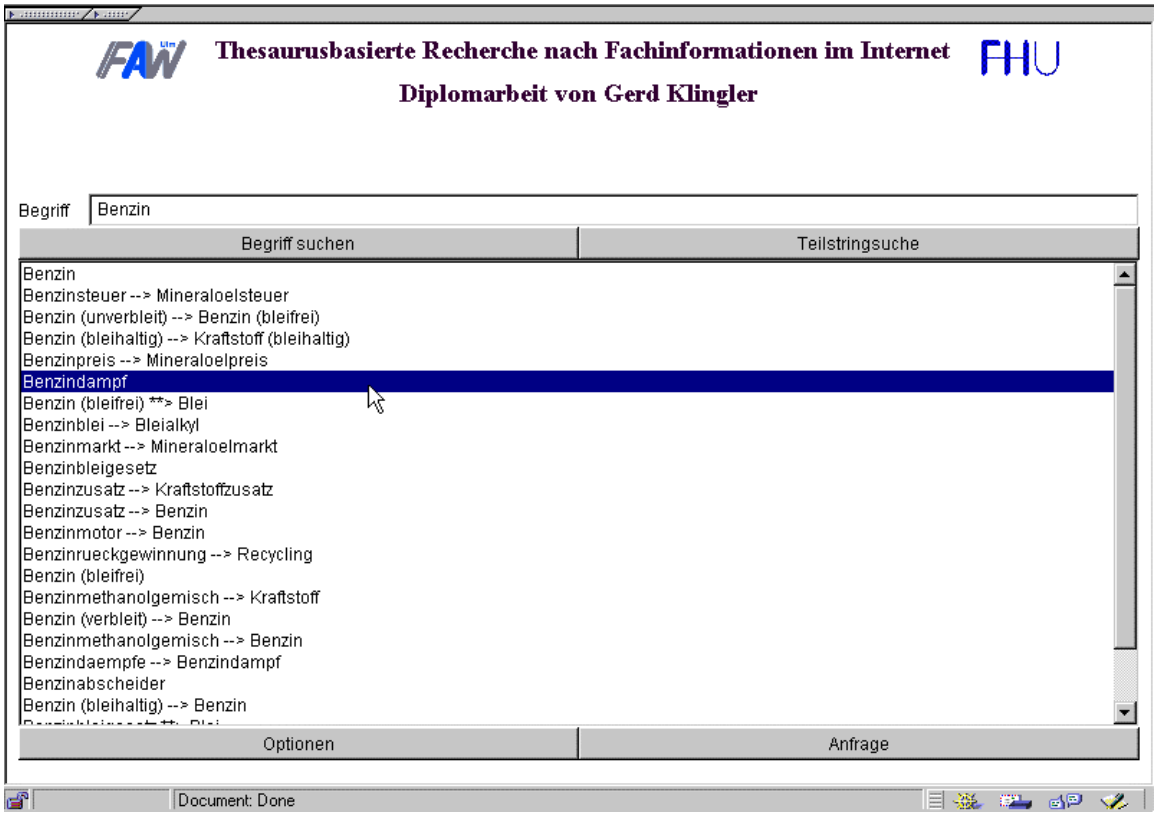

Abbildung 8.6 Ergebnis einer ersten Abfrage des Thesaurus

Wird eine Anfrage an den Thesaurus mit einem gefundenen Begriff gestartet, so liefert die Datenbankanfrage ein semantisches Netz bezüglich des gesuchten Begriffs zurück. Dabei ist die Darstellung des Netzes abhängig von den gewählten Parametern.

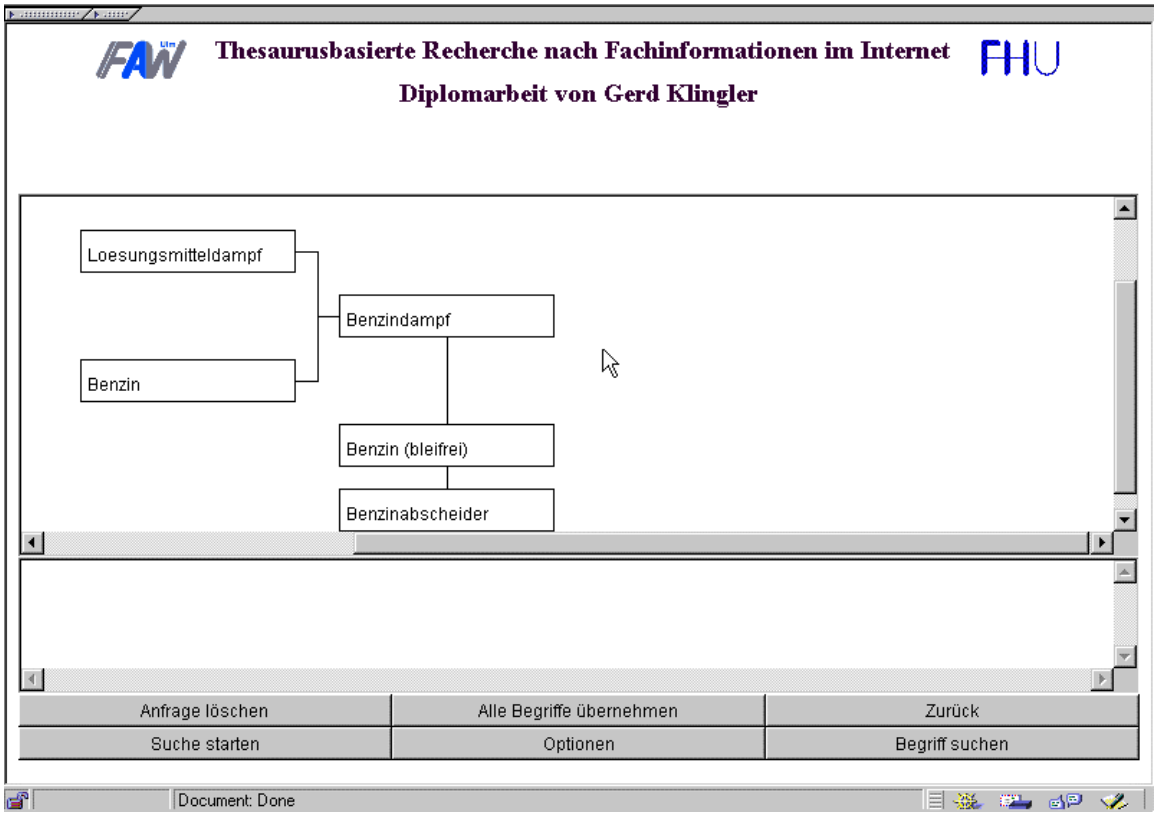

Abbildung 8.7 Netz mit je einer Hierarchiestufe für Ober- und Unterbegriffe des Terms „Benzindampf“

Anhand des Netzes kann nun weiter durch die Hierarchie des Thesaurus navigiert werden. Hierzu selektiert man den gewünschten Begriff und startet eine neue Anfrage an den Thesaurus.

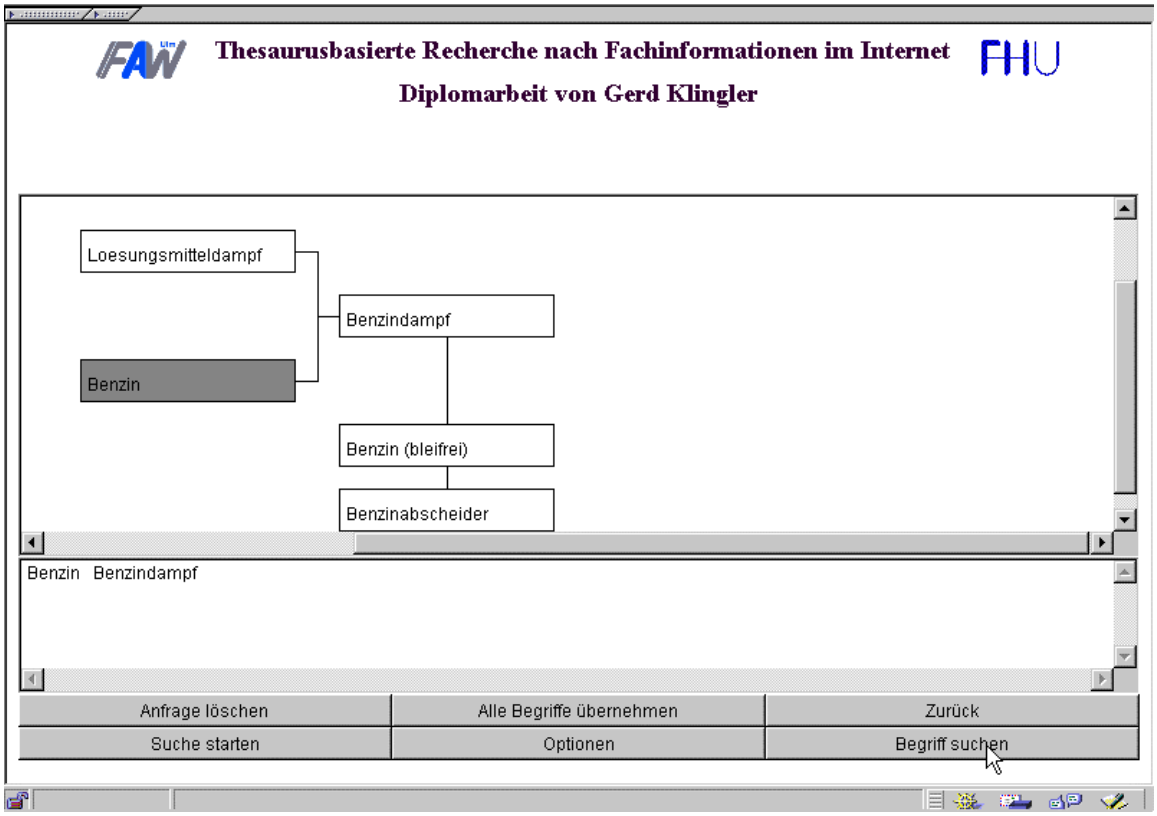

Abbildung 8.8 Erneute Abfrage mit den eingestellten Optionen und selektiertem Begriff

Als Ergebnis einer neuerlichen Anfrage erhält man dann das semantische Netz bezüglich dieses Begriff. Dabei werden die zuletzt eingestellten Parameter verwendet.

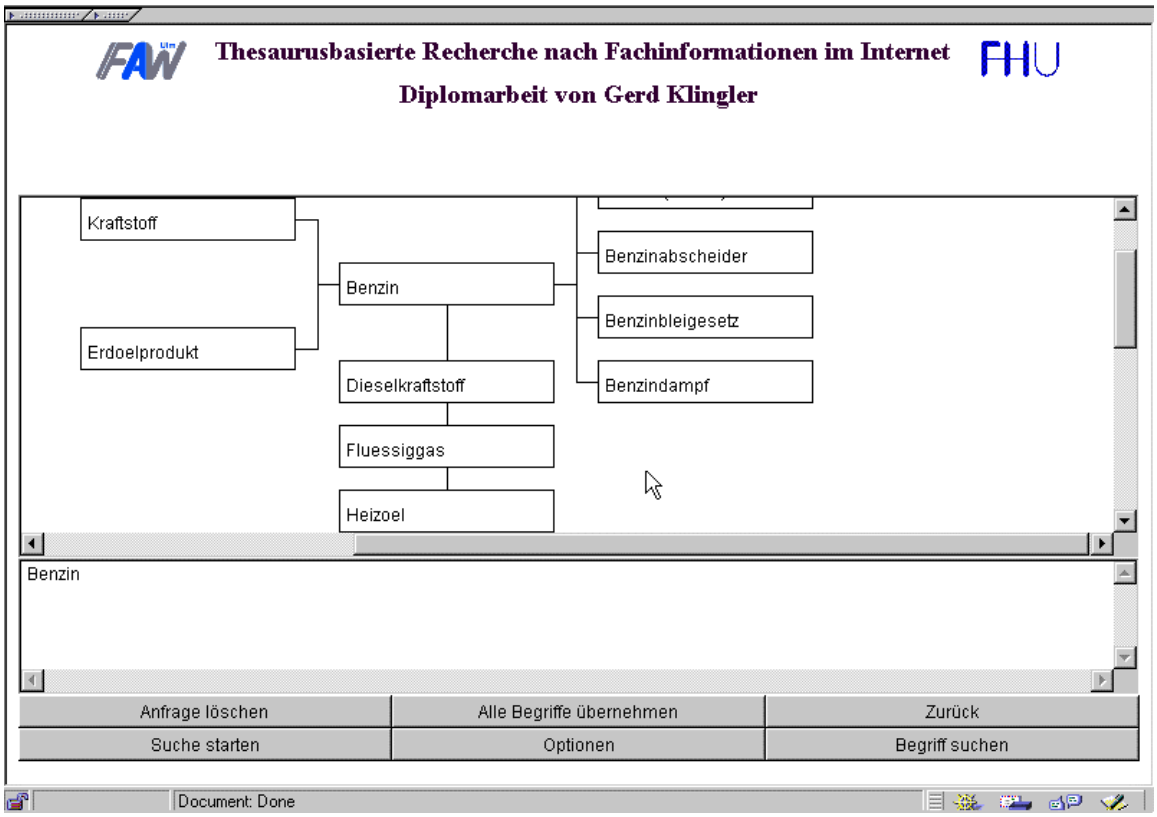

Abbildung 8.9 Netz mit je einer Hierarchiestufe für Ober- und Unterbegriffe des Terms „Benzin“

Man kann nun die Begriffe einzeln oder als gesamte Menge in die Suchanfrage für die Suchmaschine übernehmen. Dabei wird, für den Fall daß man in einer anderen Zielsprache als die aktuelle Dialogsprache suchen möchte, die jeweilige Übersetzung der Begriffe verwendet.

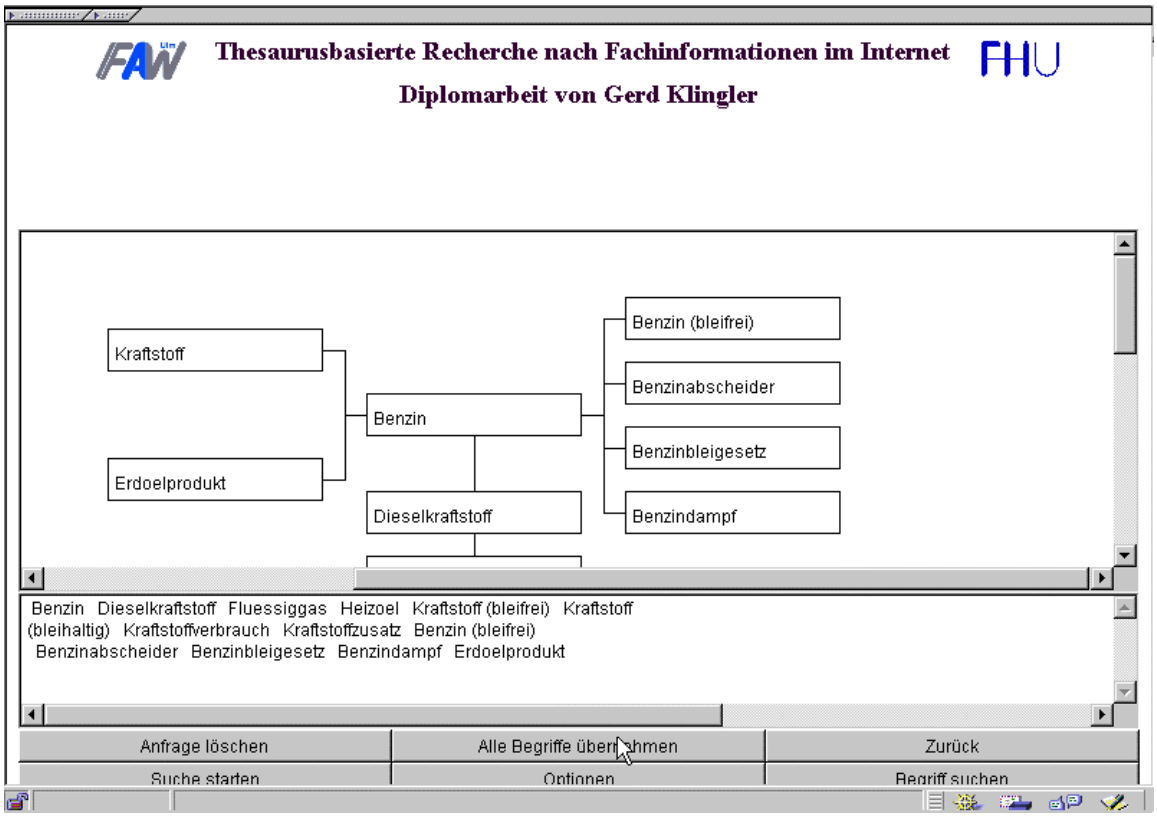

Abbildung 8.10 Übernahme aller Begriffe in die Suchanfrage

8.6 Beispiele einer Suche

8.6.1 Suche nach dem Begriff „Wurzelgemuese“

8.6.1.1 Suche ohne Thesaurusmodul

Abbildung 8.11 Suchanfrage ohne Thesaurusmodul nach „Wurzelgemuese“

8.6.1.2 Suche mit Thesaurusmodul

Abbildung 8.12 Suchanfrage mit Thesaurusmodul nach „Wurzelgemuese“

Diese Suche wurde mit folgenden Optionen durchgeführt:

Suchtiefe Oberbegriffe	0
Suchtiefe Unterbegriffe	0
Synonymbegriffe	ja
Schwesterbegriffe	nein
Verwandte Begriffe	nein
Übersetzte Begriffe	nein

Durch die Verwendung des Thesaurusmoduls wurde die Suche nicht mehr nur mit dem Begriff „Wurzelgemuese“, sondern auch mit den Synonymen „Karotte“, „Moehre“ und „Mohrruebe“ gesucht.

8.6.2 Suche nach dem Begriff „Feuerstaettenverordnung“

8.6.2.1 Suche ohne Thesaurusmodul

Abbildung 8.13 Suchanfrage ohne Thesaurusmodul nach „Feuerstaettenverordnung“

8.6.2.2 Suche mit Thesaurusmodul

Abbildung 8.14 Suchanfrage mit Thesaurusmodul nach „Feuerstaettenverordnung“

Diese Suche wurde mit folgenden Optionen durchgeführt:

Suchtiefe Oberbegriffe	1
Suchtiefe Unterbegriffe	0
Synonymbegriffe	ja
Schwesterbegriffe	nein
Verwandte Begriffe	nein
Übersetzte Begriffe	nein

Durch die Verwendung des Thesaurusmoduls wurde die Suche nicht mehr nur mit dem Begriff „Feuerstaettenverordnung“, sondern auch mit dem Synonym „Feuerverordnung“ und dem Oberbegriff „Immisionsschutzrecht“ durchgeführt.

8.6.3 Suche nach dem Begriff „Zitronensaeure“

8.6.3.1 Suche ohne Thesaurusmodul

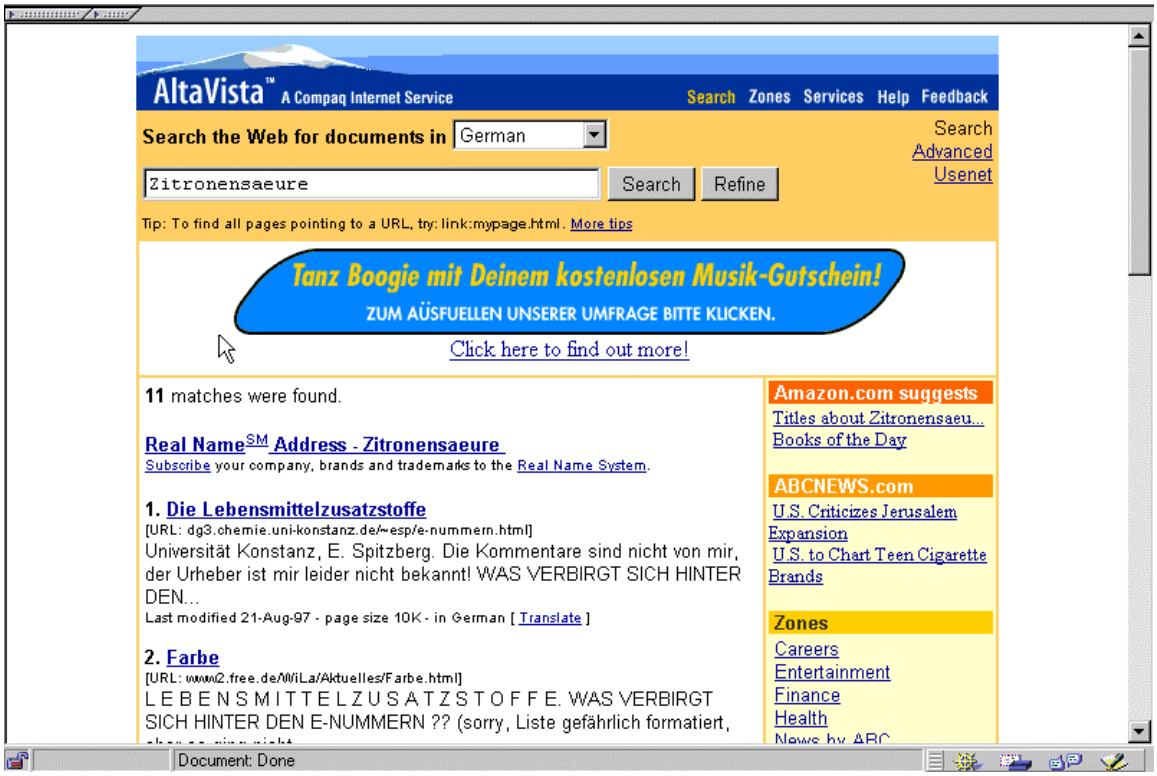

Abbildung 8.15 Suchanfrage ohne Thesaurusmodul nach „Zitronensaeure“

8.6.3.2 Suche mit Thesaurusmodul

Abbildung 8.16 Suchanfrage mit Thesaurusmodul nach „Zitronensaeure“

Diese Suche wurde mit folgenden Optionen durchgeführt:

Suchtiefe Oberbegriffe	0
Suchtiefe Unterbegriffe	1
Synonymbegriffe	ja
Schwesterbegriffe	nein
Verwandte Begriffe	ja
Übersetzte Begriffe	nein

Die Verwendung des Thesaurusmodul erweiterte hier die Anfrage nach dem Begriff „Zitronensaeure“ um das Synonym „Citronensaeure“.

8.6.4 Suche nach dem Begriff „Abwasseraufbereitung“

Wird im Thesaurus nach dem Begriff „Abwasseraufbereitung“ gesucht, so erhält man einen Verweis auf den Begriff „Abwasserbehandlung“. Abwasseraufbereitung ist in diesem Fall ein Synonym von Abwasserbehandlung. Für die Suchanfrage könnte das Thesaurusmodul jetzt je nach den gewählten Optionen eine wesentlich vielversprechendere Suchanfrage starten.

8.6.5 Suche nach dem Begriff „Schrott“

Wird im Thesaurus nach dem Begriff „Schrott“ gesucht, so erhält man neben einer Vielzahl mit „Schrott-“ beginnenden Wörtern auch „Schrott“ als Deskriptor und „Schrott“ mit dem Verweis auf „Metall“. Hier kann man dann z.B. die gewünschte Auswahl treffen und den entsprechenden Begriff für die weitere Suche verwenden.

8.7 Mehrsprachige Suche mit dem Thesaurusmodul

Man kann die Suche im Thesaurus z.B. auf deutsch durchführen und als Zielsprache für die Suche mit der Suchmaschine eine andere Sprache wählen. Es ist möglich sich das gegebene semantische Netz für die Suchanfrage in die gewünschte Zielsprache übersetzen zu lassen, oder aber die Suche im Thesaurus dann in der neuen Zielsprache vorzunehmen. Hierzu werden die Begriffe des semantischen Netzes in die Zielsprache übersetzt und in die Suchanfrage übernommen.

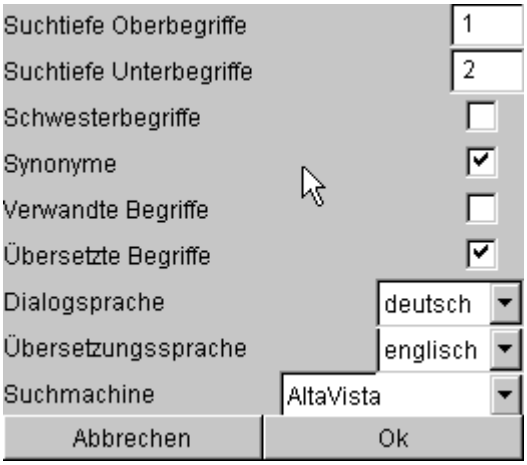

Abbildung 8.17 Mögliche Optionen für die Thesaurussuche in deutsch mit der Zielsprache englisch

Gesucht wurde im Thesaurus nach dem Begriff „Benzin“. Für die Suche wurden die in Abbildung 8.17 abgebildeten Parameter verwendet. Dieses Ergebnis wird nun für die Suche in der Zielsprache vollständig übernommen, d.h. jeder deutsche Begriff wird wenn möglich ins englische übersetzt in die Suchanfrage übernommen.

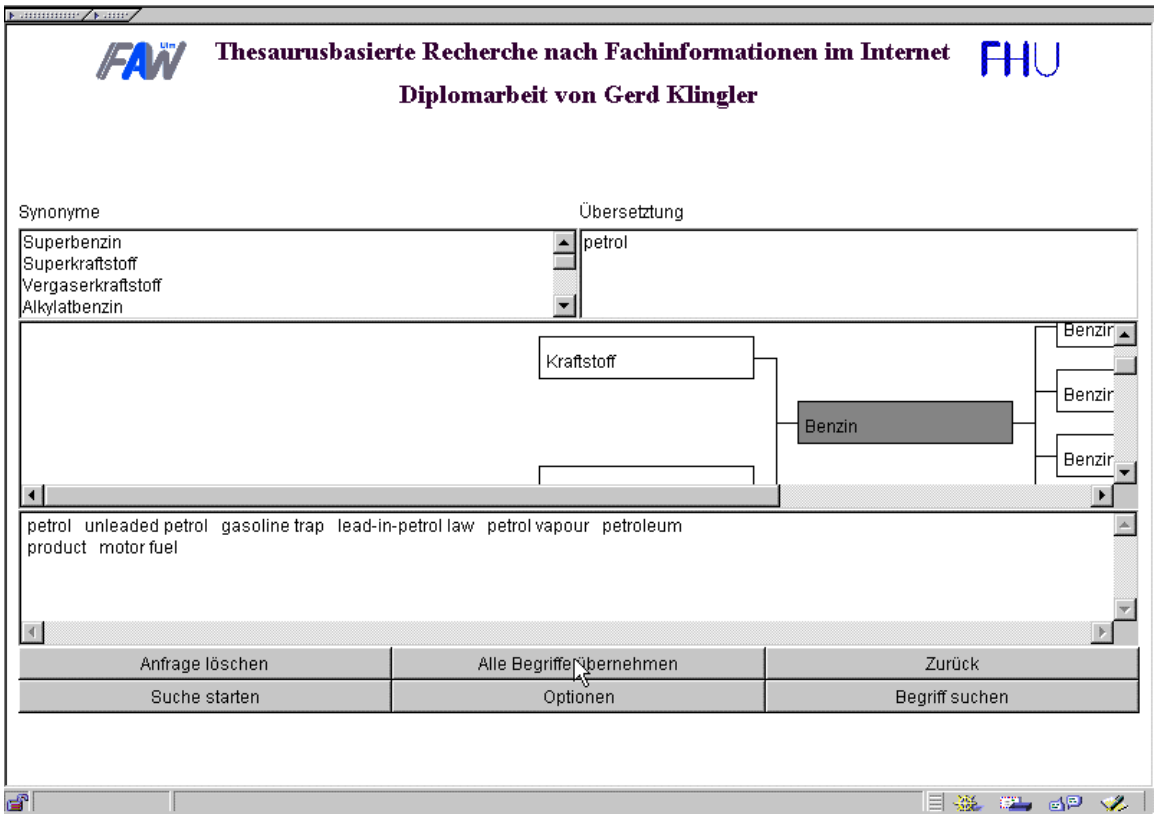

Abbildung 8.18 Übernahme der übersetzten Begriffe des semantischen Netzes

Abbildung 8.19 Suche mit Thesaurusmodul nach dem Begriff „Benzin“ in englischer Sprache

Die Suche wurde innerhalb des Thesaurus auf deutsch durchgeführt. Die vom Thesaurus erhaltenen Begriffe wurden dann in ihrer englischen Übersetzung für die Suchanfrage mit der Suchmaschine verwendet. Die Zielsprache der Suchmaschine wurde vom Thesaurusmodul aus durch die entsprechend festgelegte Übersetzungssprache bestimmt.

9 Einbindung einer eigenen Suchmaschine

Da jeder seine eigene „Lieblingssuchmaschine“ hat, ist dieses System so gestaltet, daß jeder die Möglichkeit hat seine eigene Suchmaschine zu verwenden. Allerdings muß er hierfür eine Java-Klasse implementieren.

9.1 Technische Hintergründe

Benötigt ein Java-Applet eine Klasse, so wird diese in dem Classpath des Webbrowsers gesucht. Wird die entsprechende Klasse dort nicht gefunden, so versucht das Applet die entsprechende Klasse vom Webserver des Applets zu laden. Wird die Klasse dort auch nicht gefunden, so wird ein *ClassNotFoundException* erzeugt.
Das Thesaurusmodul verwendet eine Klasse mit dem Namen *MyOwnSearchEngine*. Diese Klasse ist auf dem Webserver vorhanden, allerdings ohne richtig implementiert worden zu sein. Möchte man nun seine eigene Suchmaschine verwenden, so braucht man nur diese Klasse zu implementieren und im Classpath des Webbrowsers abzulegen.

9.2 Beispiel für eine Suchmaschine

```
package SearchModul;



/**
 * Diese Klasse soll als Beispiel dienen
 * wie man sich seine eigene Suchmaschine
 * anbindet. Dabei muß genau der Name
 * MyOwnSearchEngine verwendet werden.
 * Außerdem wird zum Compilieren auch die
 * Klasse SearchModul.SearchEngine benötigt
 * Diese Klassen werden dann compiliert und
 * im Classpath des Webbrowser im Verzeichnis
 * SearchModul gespeichert. Zur Laufzeit
 * lädt sich das Thesaurusmodul dann die
 * Klasse MyOWnSearchEngine vom lokalen System.
 *
 *
 * MyOwnSearchEngine.java
*/



public class MyOwnSearchEngine extends
SearchModul.SearchEngine {

   // Name der Suchmaschine
   // hier zum Bsp Yahoo, da der nachfolgende
   // Code die Ansteuerung von Yahoo.com vornimmt
   // in der Methode getURLString() kann durchaus
naoch
   // eine Überprüfung der gesetzten Sprache erfolgen,
   // so daß z.B. bei deutscher Sprache Yahoo.de ver-
   // wendet werden würde
```

```
   public static final String NAME = "Yahoo";



   // Vereinbarung über die verschiedenen Typen
   // von Suchmöglichkeiten
   // Typen entsprechen den AND OR * Operatoren der
   // Suchmaschine
      public final static String[] SEARCHTYPES = {"
","+", "-", "*"};

      public final static String url_address =
"http://av.yahoo.com/bin/query?p=";


   public YahooSearchEngine ( ) {
      super();
   }
   /**
    * Term zum Suchstring hinzufügen
   */
   public void addSearchTerm(java.lang.String term,
int type) {
      // neuen Term an Anfang des Suchstrings setzen
      if(search_string != null) search_string =
search_string +" "+SEARCHTYPES[type] + term.trim();
      // Term zum bisherigen Suchstring hinzufügen
      else search_string = SEARCHTYPES[type] +
term.trim();
      return;
   }

   /**
    * Namen der Suchmaschine zurückliefern
    * @return java.lang.String
   */
   public String getEngineName( ) {
      return NAME;
   }


   /**
    * aktuelle SuchString zurückliefern
    * @return java.lang.String
   */
   public String getURLString( ) {

      return url_address+parseString(search_string,
true);

   }
```

```
   /**
    * String parsen
    * Leerzeichen ersetzen, Sonderzeichen ersetzen
    * @return java.lang.String
    * @param search_string java.lang.String
   */
private String parseString(String search_string,
boolean casesensitive) {
      if (search_string == null ||
search_string.length() == 0) return null;
     String tmp = null;
     if(casesensitive) tmp =
search_string.toLowerCase();
     else tmp = search_string;

      // '+' durch '%2B" ersetzen
      while (tmp.indexOf("+") > -1) {
        tmp = tmp.substring(0, tmp.indexOf("+")) +
"%2B" + tmp.substring(tmp.indexOf("+")+1);
     }

     // Leerzeichen durch '+' ersetzten
     while (tmp.indexOf(" ") > -1) {
        tmp = tmp.substring(0, tmp.indexOf(" ")) + "+"
+ tmp.substring(tmp.indexOf(" ")+1);
     }

   return tmp;
   }
}
```

10 Abbildungsverzeichnis

11 Glossar

API:

Application Programming Interface. API bezeichnet eine normierte Schnittstelle für den Aufruf von Methoden/Funktionen einer Programmiersprache oder einer Benutzeroberfläche.

Applet:

Unter Applet versteht man in der Sprache des Internet ein besonderes Objekt, das auf einer HTML-Seite enthalten ist. Es enthält ein Programm, das in der im Internet meistverbreiteten Sprache Java geschrieben wurde. Auf der Seite kann im Bereich dieses Objekts nun z. B. eine Animation ablaufen, die nicht als Folge von Einzelbildern über das Internet geladen wird, sondern als Programm, das erst durch Ihren Web-Browser interpretiert (oder kompiliert) und ausgeführt wird. Die Anwendungen für Applets umfassen Animationen, Lauftexte, interaktive Eingaben des Anwenders, Spiele und vieles mehr.

Browser:

Ein Browser ist ein Hilfsprogramm für das World Wide Web, das z. B. zum Suchen bestimmter Informationen benutzt werden kann. Ein Browser ist in der Lage , die im Dokumentformat HTML codierten Seiten formatiert anzuzeigen.

Classpath:

Als Classpath bezeichnet man das Verzeichnis, von dem der Java-Interpreter seine Standardklassen lädt.

Frames:

Frames ermöglichen es Browserfenster in kleinere Fenster aufzuteilen. In die einzelnen Frames können verschiedene Dokumente geladen werden.

FrameSets:

FrameSets definieren die Aufteilung von Seiten in Rahmen, die jeweils eigene Inhalte mit eigenen Hintergründen, Seitenrändern usw. haben können. FrameSets werden vor allem zum Erstellen attraktiver HTML-Seiten eingesetzt.

Firewall:

Eine Firewall schützt ein dahinterliegendes Computernetz vor Angriffen aus dem Internet. Alle Zugriffe auf das interne Netz müssen erst von der Firewall genehmigt werden, bevor sie weitergeleitet werden.

FTP:

FTP steht für File Transfer Protocol (Protokoll zur Dateiübertragung) und ist das übliche Übertragungsprotokoll für Dateien im Internet. Ein FTP-Server ist ein Programm auf einem am Internet angeschlossenen Computer, der Dateien zur Übertragung mit Hilfe des FTP bereithält. Während FTP für das Übertragen und “downloaden“ von Dateien im Internet zuständig ist, dient das namensähnliche HTTP (Hypertext Transfer Protocol) dem Verbindungsaufbau und Datenaustausch zwischen WWW-Servern und WWW-Clients.

Garbage Collection (GC):

Das ist ein Prozeß, um Speicherkapazitäten von Objekten zurückzuerhalten, die nicht länger in Gebrauch sind. Ein Objekt gilt als nicht länger in Gebrauch, wenn es auf keine Referenzen mehr aus anderen Objekten im System verweisen kann und es auch keine Referenzen in lokalen Variablen oder dem Methode-Aufruf-Stack gibt.

Gateway:

Darunter versteht man im allgemeinen eine Schnittstelle zwischen zwei Kommunikationssystemen, so daß beide Systeme miteinander kommunizieren können. Ein Gateway ermöglicht z.B., daß man als Benutzer von T-Online eine Email an einen Empfänger im Internet schicken kann.

Host:

Dies ist der Rechner "am anderen Ende der Leitung", mit dem der eigene Rechner bei der Datenübertragung Kontakt aufnimmt. Der eigene Rechner wird dann als Client bezeichnet. Entsprechend spricht man von Host- und Clientprogrammen, die für die Softwareseite der Kommunikation zuständig sind.

HTML:

HTML (Hypertext Markup Language) ist eine Dokumentbeschreibungssprache, die als Dateiformat für WWW-Dokumente benutzt wird. Es integriert Text, Grafik, Videos und Sound. Der aktuelle Stand der Entwicklung ist HTML 3.2.

HTTP:

Das HyperText Transfer Protocol (Hypertext-Übertragungsprotokoll) ist das Übertragungsprotokoll von WWW-Dokumenten zwischen WWW-Servern (hosts) und WWW-Browsern (clients).

Instanz:

Ein Objekt. Wenn eine Klasse instantiiert wird, um ein Objekt zu erzeugen, dann sagen wir, daß dieses Objekt eine Instanz der Klasse ist.

Internet:

Das Internet ist ein weltumspannendes Netzwerk, das seinen Ursprung 1969 in dem Arpanet (Advanced Research Project Agency) des amerikanischen Verteidigungsministeriums hat. Später wurde das Netz für die allgemeine Nutzung freigegeben, und nachdem Netze anderer Länder an das Netz angebunden wurden, entstand der Begriff Internet. Heute besteht das Internet aus mehr als 90.000 Einzelnetzen in über 100 Ländern mit mehr als 35 Millionen Benutzern. Damit ist das Internet das größte Netzwerk der Welt.

Internet Site:

Mit Internet Site bezeichnet man einen Ort im Internet, der Informationen bereitstellt und eine eigene international registrierte IP-Adresse hat. Die meistgenutzten Internet Sites sind HTTP-Server, die z. B. Seiten im HTML-Format zum Lesen anbieten. Aber auch FTP-Server, von denen Dateien wie von einer lokalen Festplatte geladen werden können, sind Sites im Internet. Physikalisch kann man sich unter einer Internet Site einen Rechner mit Internetanschluß vorstellen, der jede Größe zwischen dem Hobbyrechner mit Modem bis zum Großrechnerverbund annehmen kann.

Intranet:

Mit Intranet werden lokale Netzwerke in Firmen bezeichnet, die zur Kommunikation TCP/IP und als Übertragungsprotokoll HTTP verwenden. Ein Intranet hat gegenüber herkömmlichen Netzwerken den Vorteil, daß der Übergang in das Internet sehr leicht möglich ist und daß die nötige Software zur Zeit meist sehr preiswert ist.

IP-Adresse:

Dies ist eine 32-Bit-Adresse im Internet, die in vier Zahlen zwischen 0 und 255 geschrieben wird. Die vier Zahlen werden durch Punkte getrennt, also z. B. 123.234.56.78. Jeder Benutzer hat solch eine Internet-Protokoll-Adresse. Beim Zugang über PPP oder SLIP wird die IP-Adresse meist nur für die Dauer der Verbindung dynamisch zugewiesen. Da diese Adressen schlecht zu merken sind, spricht man die Server im Internet fast immer mit ihren Namen an. Ein Rechner, der Nameserver genannt wird, sorgt dafür, daß aus einer von ihm verwalteten Liste zu dem Namen die korrekte IP-Adresse eingesetzt wird.

JDBC:

Java Database Connectivity stellt eine Schnittstelle zwischen Java-Programmen und Datenbanken her. Es ist angelehnt an ODBC.

JDK (Java Developer Kit):

Das ist ein Softwarepaket, das von SUN Microsystems an Java-Entwickler verteilt wird. Es enthält einen Java-Interpreter, Java-Klassen und Java-Enwicklungstools.

ODBC:

Open Database Connectivity ist der De-facto-Standard einer datenbankunabhängigen API. ODBC-Treiber sind für nahezu alle Datenbanken der Windows- und Macintosh-Plattformen vorhanden. Viele Anbieter von UNIX-Datenbanken bieten auch ODBC-Treiber an.

Server:

Ein Server (wörtlich:Diener) ist in lokalen Netzwerken ein Computer, der anderen Computern Daten, Programme u.s.w. zur Verfügung stellt.

Schnittstelle:

Die Schnittstelle eines Systems ist die Zusammenfassung aller von außen benötigten und aller von außen abrufbaren Größen, sowie allgemeiner Informationen für die Verwendung des Systems. Zugleich umfaßt sie Vereinbarungen, sog. Protokolle, über die Art und Weise, wie Informationen ausgetauscht werden.

Thread:

Ein einzelner unabhängiger Ausführungsstrom innerhalb eines Programmes. Da Java eine multithread-fähige Programmiersprache ist, kann mehr als ein Thread innerhalb des Java-Interpreters zur gleichen Zeit ablaufen.

TCP/IP:

Diese Abkürzung steht für Transmission Control Protocol / Internet Protocol. TCP ist für den Auf- und Abbau von Verbindungen zwischen allen Computern in einem Netzwerk zuständig. Es steuert den Datenfluß im Netz und sorgt für eine vollständige Datenübertragung. IP übernimmt die Organisation und Adressierung der Daten. Für die Übertragung werden die Daten in Pakete unterteilt und beim Empfänger wieder zusammengefügt. Dieses Protokoll wird sowohl in lokalen Netzwerken als auch im Internet verwendet.

URL:

Der Uniform Resource Locator (URL) stellt die Adresse eines Dokuments oder eines Servers im Internet dar.
Der allgemeine Aufbau eines URL ist je nach Typ unterschiedlich, entspricht aber der Form Dienst://Hostname:Port/Pfad/Seite#Marke, wovon nicht immer alle Elemente angegeben werden müssen. Ein URL kann unter anderem eine FTP-, eine WWW- (HTTP-), eine File- oder eine E-Mail-Adresse sein.

Webserver:

Ein Webserver ist ein mit dem Internet verbundener Rechner mit einem Programm, das dafür geeignet ist, WWW-Dokumente zur Anzeige und/oder Dateien zum Download bereitzustellen.

WWW:

Diese Abkürzung steht für Word-wide Web, also weltweites Netz. Das WWW stellt eine Art "Unternetz" des Internet dar, das von WWW-Servern gebildet wird. Diese Server stellen Daten im HTML-Format zum Abruf bereit. Das WWW, auch W3 genannt, bietet damit die Möglichkeit, Textinformationen, Grafiken, Töne sowie Animationen im Internet zu übertragen.

12 Literaturhinweise

- Dicken Hans, JDBC Internet-Datenbankanbindung mit Java, International Thomson Publishing Company, 1997
- Flanagan, David, *Java in a Nutshell*, O'Reilly, 1996
- Misgeld, Wolfgang D. / Gruber, Jürgen: *Einführung in Java*, Hanser, 1996
- Newman, Alexander u.a., *Java: Referenz und Anwendung*, Que, 1997